Improving the educational system for (future) fingerprint experts with the use of researched study methods and e-learning.

University of Twente

Creative Technology

EEMCS

Desly Elskamp

S2013800

Supervisor: Dr. Job Zwiers

Critical Observer: Caroline Gibb

02/07/2021

Abstract

Fingerprints have been used as a tool to establish a connection between a trace recovered from a crime scene and an individual for over a century. For a fingerprint examiner to achieve an expert status, they must pass a series of extensive examinations throughout their training. Successfully passing these exams shows that they posses specialised knowledge greater than that of a lay person. However, the current educational system is outdated, resulting in students spending many years on the process or even dropping out because of a lack of perspective and motivation.

This research created the first step in designing a new fingerprint comparison training tool to adapt the educational system to the 21st century. A literature research was performed on researched study methods and e-learning, followed by an experiment using two experimental feedback types: '*Task feedback'* and '*Peer feedback'*. The two experimental feedback types were studied and compared to learning without feedback, or very little feedback to mirror current industry practises. The experiment showed some interesting results and behaviour in the different feedback groups and should be further developed and explored with fingerprint examiners.

Acknowledgements

First of all, I would like to thank my supervisor Job Zwiers for his supervision, keeping me on track and all the feedback during our weekly sessions, which helped me developing my ideas and prototypes.

Secondly, I must thank Caroline Gibb for all her help in gaining insight in the forensic domain. She has been part of forensic departments for many years and she is now working on her PhD on this exact topic. She gave me a huge head start by providing research papers and sharing all her knowledge and experience. She made me aware of all the standards in fingerprint comparison and her feedback during the meetings and on my work pushed it to the next level. Her endless enthusiasm was also a great motivational boost!

Finally, I would like to thank all the people participating in the experiment. The experiment was no quick act and without their help I would not have been able to finish this research.

Index

Abstract 2
Acknowledgements 3
1. Introduction 6
1.1 Background 6
1.2 Challenge 6
1.3 Research questions 7
1.4 Report structure 7
2. Literature research 8
2.1 Researched study methods relevant to perceptual expertise 8
2.2 Feedback types and their influence on learning 9
2.3 Advantages and disadvantages of e-learning 9
2.4 State of the art 10
2.5 Literature research conclusion 11
2.6 Limitations 12
2.7 Further research 12
3. Realisation 13
3.1 Feedback ideas 13
3.2 Interfaces 15
3.2.1 Introduction screens 15
3.2.2 Analyses 15
3.2.3 Comparison 16
4. The experiment 17
4.1 The application 17
4.2 Implemented feedback 19
4.2.1 Current Feedback 19
4.2.2 Task Feedback 20
4.2.3 Peer Feedback 20
4.3 Survey 20
4.3 Participants 20
4.4 Processing data 20
5. Experiment results 22
5.1 No Feedback & Current Feedback 23
5.2 Task Feedback 23
5.3 Peer Feedback 23
6. Discussion 24

6.1 Conclusion 24
6.2 Limitations 24
6.3 Future work 25
References 26
Appendices 28
Appendix A 28
Appendix B 29
Appendix C 58
Appendix D 61
Appendix E 62

1. Introduction

This chapter will briefly describe the problems that arise during the process of becoming a forensic fingerprint expert and how these problems should be tackled. After giving more information about the background the objectives will be explained, followed up by the research questions and finally an overview of the work that has been done.

1.1 Background

The process of fingerprint (latent print) comparison has been an important tool to identify criminals for over a century [1]. Latent print analysts often serve as witnesses in serious criminal trials and therefore have a huge influence on the lives of individuals who face trial. To acquire an expert status as a latent print analyst, students have to follow an extensive course and successfully pass a series of intensive exams [2][3]. After successfully passing the exams the analyst is rewarded with a special certification, which implies that the individual has achieved a level of specialized knowledge greater than that of a novice, based on their training, study and experience. In the criminal justice system, this certification will be submitted to the court to assist the judge in determining whether the expert witness is qualified to offer opinion evidence in court and that their testimony is valid and reliable.

During the process of becoming a fingerprint comparison expert several problems often arise; Firstly, latent print comparison has been viewed as an infallible craft from the beginning of its inception in the early 1900s. For over a century, fingerprint expert witness testimony has been accepted by the courts with minimal challenges to the reliability of the evidence. However, a lack of adherence to the scientific method has since been exposed to a result of high-profile erroneous identifications. As a result, scientific theory and methodology are slowly becoming a part of this domain. This has resulted in outdated educational methods within law enforcement agencies and not at the level it should be. Secondly, examiners often get a few months of excessive training during their initial induction phase and are then released out in the field and are expected to develop their skills simply by studying thousands of fingerprints often with a lack of mentoring and usually without getting useful feedback. Research in this domain shows that most of the progress takes place in the first three months of training, but from there the progress stagnates [4], causing some students to spend up to ten years before reaching an expert status. The progress students make is not extensively tracked either, resulting in situations where both the student and the trainer do not have a clear vision on the actual skill level, or developmental progress of each individual. The current educational environment results in many talented students leaving the industry due to a lack of perspective and motivation.

1.2 Challenge

To retain (future) fingerprint experts the first step is to provide a tool with accurate empirical capture, which is achieved through digitization. This places accountability on both the trainer and the student and gives supervisors an accurate and objective portrayal of individual progress. Secondly, we increase user motivation by utilizing a fit-for-purpose e-learning platform, making use of researched study methods with various types of feedback. Thirdly, by empirically tracking individual progress, students will have a clear overview of their learning curve, whilst also providing them with a trustworthy perspective on their future progress, including a reliable record of their training, skills and experience.

The goal of this study is to design an online e-learning platform that meets these requirements. In addition to implementing researched study methods, the online platform will make use of feedback. There are several feedback types that can be applied (for example, direct feedback, block feedback, task feedback or process feedback) and this study will investigate the most effective, and based on user feedback, the most user friendly.

As mentioned earlier, tracking the progress of students is also an important feature of the online tool. Empirical data, such as the time it takes to finish a comparison, or the number of errors that are made, will be collected in order to provide useful feedback and to enable supervisors to have a more demonstratable and objective view of the skill levels of students. By empirically tracking individual progress, students will have a clear overview of their learning curve, whilst also providing them with a trustworthy perspective on their future process.

1.3 Research questions

- How can we improve the educational system for (future) fingerprint experts with the use of researched study methods and e-learning?
    - Which training protocols or methods have been proved to be (un)successful for training perceptual expertise and why?
    - What are important (dis)advantage of e-learning and how does e-learning influence results compared to offline learning?
    - How do different types of feedback influence the learning process compared to learning without feedback?

1.4 Report structure

The structure of this report will be as follows; First of all, a literature research will be done to answer the sub-questions. Then, the realisation phase will be explained in which two prototypes will be discussed. Thereafter, the experiment will be explained and the results of this experiment will be discussed. Lastly, the conclusion and discussion about the study will be given.

2. Literature research

To answer the sub-questions, a literature review was conducted to investigate different study methods relevant to the development of perceptual expertise. Then the focus will be moved to the different types of feedback, because of feedback being the most well-known and widely used study method. Thereafter, the research discusses the main advantages and disadvantages of e-learning to consider when creating an online platform. Then, a short overview of the current state of the art fingerprint educational tools will be given. Finally, a conclusion will be drawn from the literature research to give the reader a clear overview of what is needed to improve the education of (future) fingerprint experts.

2.1 Researched study methods relevant to perceptual expertise

Despite the limited number of studies on this topic, several methods have been found that appear to be effective for training perceptual expertise. The most common and often named method is the usage of feedback. White et al. [5] claim that providing feedback can reliably improve the accuracy of participants trying to match unfamiliar faces. They state that short feedback sessions incorporated in daily routines could already be very beneficial for people who require this skill in their job. The significant improvement that feedback gives is substantiated by Searston and Tangen [4]. They show that participants who received feedback showed significantly better results than those who did not receive any feedback. Their experiment shows that when individuals compare fingerprints with feedback, their accuracy is higher compared to individuals that compare fingerprints without feedback (82.23% and 67.00% respectively). In addition, Herzog and Fahle [6] show that receiving correct feedback under the right conditions indeed leads to significantly better results than receiving manipulated or no feedback. They state that correct feedback also increases the speed of learning and reduces the interindividual differences. These three studies clearly suggest that feedback leads to improvement during the process of learning perceptual expertise.

Whilst the usage of feedback seems to be beneficial, not everyone agrees on its significance in the perceptual domain. Herzog and Fahle [6] show that performance was significantly better with correct feedback, yet on the other hand they mention that the positive effect of feedback is neither very specific nor accurate, because reduced feedback did not change the results dramatically compared to complete feedback. Another study on perceptual learning in non-stationary contexts shows that although earlier observations suggest that feedback may increase the efficiency of learning, perceptual learning without feedback does not underperform compared to perceptual learning with feedback [7]. An explanation for these results may be that different situations might require different types of feedback to show significant improvement. Therefore, different types of feedback and their influences on learning processes will be further discussed.

Besides the usage of feedback there are three other methods that seems to have a positive influence on learning perceptual expertise. Firstly, Searston and Tangen [4] show that using the two training protocols, specifically labels and contrast, leads to a significantly greater ability to discriminate new pairs of fingerprints compared to the baseline training protocol. The labels and contrast methods both aim to make people study the details of fingerprints rather than the overall image or simple inequalities. Both methods showed significantly better results compared to the baseline test, where 67.00% of the given answers were correct, with 79.46% correct decisions on the labels test and 76.79% on the contrast test. The third protocol has the same intentions as the other two methods. Popov and Reder [8] state that although it is logical that finding a target among very similar distractors is more difficult than finding one among only partly similar distractors and therefore leads to more errors, this is a short-term effect and will lead to better results in the long term. This is because people tend to use partial matching, which means they base their comparison on a few clear characteristics. By using very similar distractors people are forced to study the material in more detail which leads to a significant improvement in skill on the long term [8]. These two studies explain the importance of studying perceptual expertise on detail level rather than using partial matching. Challenging people by presenting highly similar fingerprints and making them label features and look for differences can thus be difficult for novices but might lead to better results on the long term.

2.2 Feedback types and their influence on learning

Earlier paragraphs explained that feedback can be a very efficient study method. However, not all researchers agree on the significance of the results compared to learning without feedback. One reason for this could be that different types of feedback lead to different results in different situations. Herzog and Fahle [6] describe different situations in which participants received feedback. As might be expected, uncorrelated or incorrect feedback can slow down the learning progress, or even decrease the skill level and this should of course be avoided. However, more interesting is the claim that reduced feedback does not immediately lead to a significant decrease of progress. In their experiment the difference between people who received an error message after every mistake and people who received the number of correct answers after every block of 80 exercises was not significant, although the results of the block feedback were slightly better. Even the test where participants got an error message after only 50% of their mistakes still showed a significant improvement compared to the participants who received no feedback [6]. This research shows that giving more direct feedback does not always lead to better results in the long term and should therefore be tested in different situations.

In addition to the amount of feedback that is given, there are also four different types of feedback that can be useful in different situations. The first type of feedback is directly related to the given answer (e.g., "You need to include more about minutiae in your answer"). This type of feedback could be useful when the user makes incorrect interpretations. The main disadvantage of this type of feedback is that is does not provide further information for students who do not have enough understanding of the topic. When this type of feedback is used too often it might also cause students to focus more on the sub-goal than on the overall understanding of the process [9].

When there is a need for more information, it is possible to use the second type of feedback which focusses more on the process leading to the answer (e.g., "This paragraph might make more sense if you use the strategies that we discussed during the lectures.") Hattie and Timperley [9] state that this type of feedback appears to be more effective than the first one for enhancing deeper learning. The last two types of feedback are more focused on the person giving the answer (e.g., "You are a great student!", or "You already know the importance of documenting your analysis, check to see if you incorporated it") and are rarely as effective as the first two types. Hattie and Timperly [9] claim that the first two methods can be very effective when used in the right situation, but other research states that the results of both methods are not significantly better than the results without receiving feedback [10]. Therefore, the significant influence that feedback might have will still be arguable and should be tested in the specific situation where it will be used.

2.3 Advantages and disadvantages of e-learning

Since feedback and other methods have to be personalized for every student, it could be beneficial to make use of e-learning. The role of e-learning in the educational system has been growing rapidly during the last decade [14] and several studies provide benefits and advantages derived from the adoption of e-learning technologies. The most common advantage mentioned is that e-learning can always take into consideration the individual learners differences. In contrast to face-to-face learning there is more attention paid to individuals and users are able to follow education in their own tempo and on their own level [11]. Biedermann et al. [12] explain that this option of self-directed learning also enables users to study whenever and wherever they want to, which can be crucial for people with full-time jobs like fingerprint examiners who have already achieved the expert status.

E-learning also creates the opportunity to engage with other academics or experts which allows them to discuss their work on forums. Since feedback seems to be an important method for perceptual learning this can be used by giving users the option to receive peer-feedback [12]. E-learning also enhances the ease of accessing huge amounts of information. This is one of the advantages that leads to more satisfaction under users of e-learning. Liu [13] shows that users of e-learning were more satisfied with the learning materials and learning environment than those who used face-to-face education. The improved satisfaction towards the learning environment could be because of the lowered barriers of reaching out to others that some people often experience.

Although many users are more satisfied when using e-learning and e-learning can also lead to better results [13], there are two main disadvantages of e-learning that should be considered. Firstly, the lack of personal contact can make people feel distanced and therefore e-learning requires a higher motivation and time management [11]. However, Dominguez et al. [15] explain that the motivation of users can possibly be improved by adding gaming elements like reward systems and competitive mechanisms. Secondly, e-learning might also not be useful in all disciplines. Theoretical disciplines can easily implement e-learning, while it might be less effectives for studies where practical skills are necessary [11]. The effectiveness of e-learning thus depends on personal motivation and the domain in which it is used.

2.4 State of the art

Currently there are no widely used fingerprint comparison training tools. Current available tools have been primarily developed to capture, and document examiner reasoning with minimal focus on educational capability. There are several tools that make it easier for examiners to mark minutiae, but those do not show any possibilities of receiving feedback or sharing knowledge based on researched study methods. To be seen as a training tool, it was decided that any features like feedback or other study methods must be present. Therefore, there will be just two fingerprint comparison training tools discussed in the state of the art. Notably there are tools which are designed for professional use that could be useful during the training phase. Those tools will be described and summarized as well.

ACEware

ACEware [16], is designed to create a link between students and experts. Students will first use the system to analyse the fingerprint to say if the mark is suitable for a comparison or not. This is performed by marking all the available minutiae. If the fingerprint is suitable for comparison the student will perform a comparison and make one of the three following judgements as to the source of the fingerprint: identification, inconclusive or exclusion. This data is saved and will be compared with the data from an expert and/or database with the use of an overlay (showing all the data on one fingerprint image simultaneously). By comparing the data, the user can see which minutiae he noted correctly, which minutiae he noted incorrectly and which minutiae he completely missed. By creating groups, experts and teachers can use this software to create exercises for students and take data from students to discuss with the rest of the group. Because of the direct link between experts and students, it is easier to give useful feedback to the user.

PiAnOs

PiAnOs [17] is quite similar to the ACEware tool. It allows users to mark the minutiae on the fingerprint and can create an overlay after the user is finished with the comparison. The difference between the PiAnOs software and the ACEware tool is that PiAnOs is designed for individual use. There are no features that allow direct communication between users and experts and the overlay which is created is taken from a database, which is filled with earlier results from experts. The feedback that is received is therefore focused on the person and requires self-checking.

AFIS

AFIS stands for "Automated Fingerprint Identification System" and is designed to store fingerprints and find highly similar fingerprints in databases in a relatively short period of time [18]. Unfortunately, the system is not yet precise enough to identify or exclude highly similar fingerprints without human intervention. This is why fingerprint experts are still required to check if highly similar fingerprints found on the crime scene can be attributed to an individual's fingerprint stored on the system. However, for non-complex fingerprints, which are used to train students, AFIS generally indicate if it is a match or not. Therefore, AFIS could be used by students to check if the system draws the same conclusion as they did, which can be described as receiving task feedback.

LQMetric

LQMetric [19] is a Latent fingerprint quality metric used to rate the quality of fingerprints. When a fingerprint is analysed by the algorithm, different zones of the fingerprint will be color-coded. The different colours red, yellow, green and blue represent the clarity of the ridges and show which zones will likely present the most reliable information (a red colour refers to unclear zones, yellow to semi-clear zones, green to clear zones and blue to perfectly clear zones). Again, this software tool is not designed to be used by students, but a system like this could be useful to provide students with process feedback. When users of the newly designed training tool do not find enough minutiae, this algorithm could show them where more information could possibly be found.

2.5 Literature research conclusion

This research has highlighted several options that could lead to significant improvements within the current fingerprint educational system. Although not everyone agrees on the significant impact of feedback, it appears both students and experts could benefit from learning with feedback compared to learning without feedback. What is promising from these studies, is that several show a significant improvement in results when using feedback, whilst other studies show that correct feedback did not slow down or hold back learning. The two types of feedback which showed significant improvement should be applied under the right conditions to improve the education of (future) fingerprint experts. The first type directly focusses on the given answer and is most effective when there are false interpretations of the theory. If there is a lack of understanding the feedback should focus more on the process leading towards the answer than the answer itself. Other methods focused on making users more aware of the details of the fingerprints while comparing images rather than making them use partial matching. In the future, this sub-goal could be reached by using the aforementioned methods described in earlier paragraphs, or new methods could be created which lead to the same result.

E-learning brings a list of advantages that should be implemented to improve the fingerprint education. It can be said that individualization is one of the major advantages of e-learning. To make use of e-learning in the best way possible, the individualization should be centralized to enable users to work on their own pace and skill level and to provide them with useful, relevant feedback. Additionally, the option of peer-feedback and forums is very promising and could be implemented in the e-learning environment to enable users to exchange knowledge. Other advantages of e-learning include studying wherever you want, studying at your own pace, increased knowledge exchange and fairer and more objective assessments.

It can be concluded that, although there is still a lack of information on study methods in learning fingerprint comparison, software tools are falling behind on using the current knowledge. There are some tools which simplify noting minutiae on fingerprint images, but there are only two tools which enable some kind of feedback. Current fingerprint comparison tools do not show any other study methods next to peer-feedback and task feedback and the progress of users is not tracked either. To improve the software tools, more study methods should be implemented and should be designed for students rather than experienced users.

2.6 Limitations

Although some studies show promising results regarding feedback and other study methods for learning perceptual expertise, not all authors agree on its significance compared to learning without feedback. The reason for this could be the fact that there can be huge differences between the actual tasks that need to be done. For example, developing visual expertise in birdwatching could show significant improvements when feedback is used, while in fingerprint comparison this could be different. A lot of research that has been done in the perceptual expertise domain could therefore show different results for fingerprint comparison. The main limitation when considering this topic is that there are just a few studies that focus on studying fingerprint comparison rather than training perceptual expertise. The single study found shows that feedback has a significant influence on development, however the types of feedback are not discussed.

2.7 Further research

Because of the lack of information more research on study methods for training fingerprint comparison is crucial to improve the education of (future) fingerprint experts. Feedback in the training of perceptual expertise has shown promising results and therefore feedback will be the focus point for the remainder of the paper. An experiment on the use of feedback will be discussed, where experimental feedback types will be compared to learning without feedback and with the current practise, where analysts only get feedback on their final opinion. This experiment will offer recommendations on which feedback protocols should be further tested and implemented in future learning environments to improve the education of (future) fingerprint experts.

3. Realisation

This section will discuss all the feedback ideas and the prototypes that were created during the realisation phase. Firstly, the list of feedback ideas will be given and all ideas will be explained separately with the help of visuals taken from the first prototype. Secondly, the second prototype will be shown, which visualizes the interfaces for the experiment. Both prototypes were created in adobe illustrator and replace the usual paper prototypes. This was done since social contact was discouraged and all meetings were held online.

3.1 Feedback ideas

The time limit only allows for a few feedback types to be tested. However, to make a well-considered selection of experimental feedback types a list of ideas was created. The list consisted out of the following feedback ideas:

No feedback

The participants in this group will get no feedback at all and will be used as the control group. This entails that the results from the other feedback types will be compared to the 'no feedback' results to see if there is a significant difference.

Self-regulatory feedback

The participants in this group will have to compare the fingerprint pairs without direct feedback, but after every completed pair they will receive an overlay. This overlay shows them a high-quality comparison done by an analyst with an expert status, which enables the participant to develop their skills by comparing their work to the overlay. The overlay will show all the noted minutiae as well as the final decision of match / no match as can be seen in figure 3.1. When this feedback type would be implemented the participant should be able to hide and show the overlay.

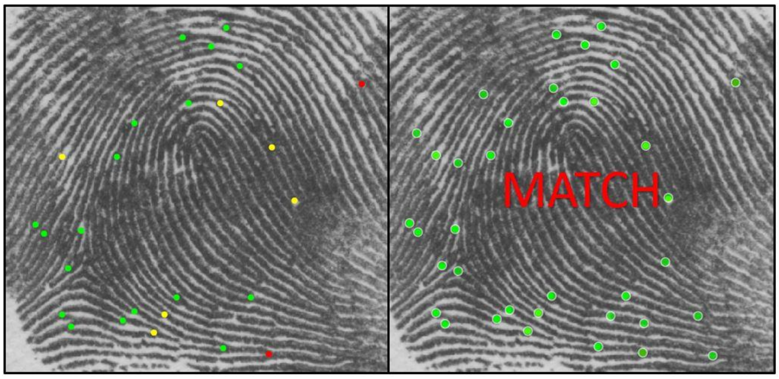


Figure 3.1: Self-regulatory feedback

Task feedback 1

The participants in this group will get direct feedback when they mark minutiae, telling them if they did it correctly or incorrectly. If they mark the minutiae on the correct place and with the correct orientation and label, they will get a green flash around their marking. When their marking is semi-correct, on the right position, but with an incorrect label or orientation, the flash will be yellow and when the marking is incorrect the flash will be red. The application will also show the participant if their final decision was correct or incorrect. The flashes that will indicate the correctness of the marked minutiae are visualized in figure 3.2. The flashes are semi-transparent to make sure participants can still see the entire fingerprint. The flash will slowly decrease in size as well, so that it will disappear from the screen without it being likely to be missed by the participant.

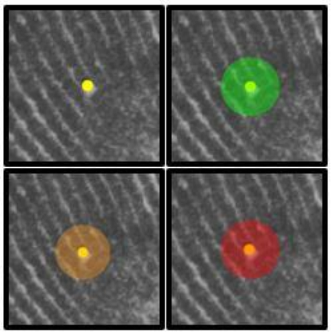


Figure 3.2: Task Feedback 1

Task feedback 2
The participants in this group will have to compare a fingerprint pair without the use of direct feedback, but when they finished a pair they will get informed if their final decision was correct or incorrect. No further information will be given to the user. This is basically the current practise and thus could be interesting to test against the no feedback group and experimental groups. When a green screen pops up, as shown in figure 3.3, the participant would be able to continue to the next page. When the given answer is incorrect, the pop-up will be red and the participant will be allowed to try again.

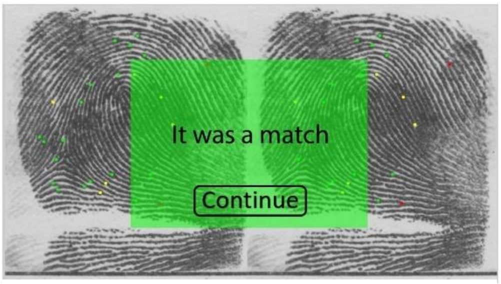

It was a match

Figure 3.3: Task Feedback 2

Process feedback 1
The participants in this group will be shown in which area they should look for more information. A pop-up will appear when the user wants to continue to the next step while they did not mark enough minutiae yet. The pop-up would show an image with colour coded areas to show where most of the information can be found (figure 3.4). A green area refers to a very clear area where participants could possibly find most of the minutiae. A red area suggests that the ridges in this area are too vague to find any reliable minutiae and yellow areas are somewhere in between the other two.

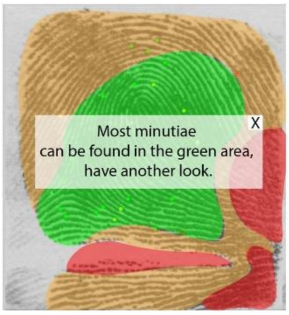
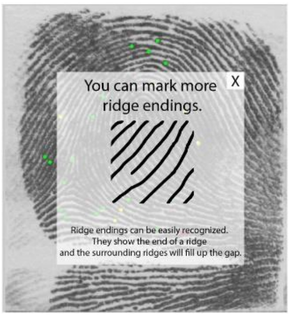

Most minutiae
can be found in the green area,
have another look.

Figure 3.4: Process Feedback

Process feedback 2
The participants in this group will be shown which type of minutiae they still miss in their analysis. A pop-up will appear when the user wants to continue to the next step while they did not mark enough minutiae yet, or when they might have made more than X mistakes with a certain minutiae type. The pop-up will show an image of the minutia as an example and a very short explanation as shown in figure 3.5. With this information the user will be able to focus on this minutiae and will hopefully find more information in the fingerprint.


You can mark more
ridge endings.

Ridge endings can be easily recognized.
They show the end of a ridge
and the surrounding ridges will fill up the gap.

Figure 3.5: Process Feedback

Peer feedback
Participants in this group will be asked to give feedback to an existing comparison next to receiving feedback on their own work. Participants will compare one pair themselves and will receive peer-feedback. After that, they will be asked to give feedback on one comparison done by someone else. It will be interesting to see if experiencing two different points of view leads to more insight. A point of attention is that peer feedback is usually not received directly, but possibly after a few days.

After a discussion with the expert and supervisor it was decided to pick two experimental feedback types to compare them with the current practise, which is 'Task Feedback 2', and learning without feedback. Firstly, it would be interesting to see if there is any difference in results between learning without feedback and the current practise. And secondly, it will be interesting to see if one of the two experimental feedback types show any significant improvement compared to the current practise and learning without feedback. The two experimental feedback types that were picked from the list are 'Task Feedback 1' and 'Peer Feedback'. 'Task Feedback 1' was picked because it is expected that this feedback type could have a huge influence on developing the basic fingerprint comparison skills, since misunderstandings and small mistakes should quickly be solved. 'Peer Feedback' was picked as the second experimental feedback type because of its two points of view. Participants will not only have to do their own analyses, but will also be introduced to critically reviewing others work, which is very useful in the fingerprint comparison domain.

3.2 Interfaces

3.2.1 Introduction screens

The second prototype focusses on the user interfaces and the functionalities that should be implemented. The prototype also visualizes the style that will be used for the final application.

When participants start the application a welcome screen will be launched. On this welcome screen participants should be able to enter a username before they continue. The welcome screen is visualized in figure 3.6.

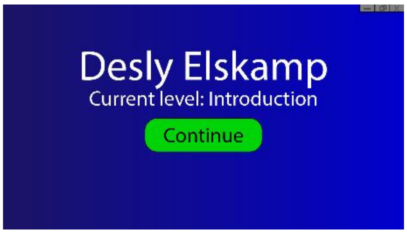


Figure 3.6: Welcome screen

When participants click "continue" they will be sent to the page with an explanatory video. This video will provide the participants with information about how fingerprint analyses and comparisons work as well as how to control the application. Participants can watch the video multiple times if they want to and can make jumps in the video by clicking the timeline. It should also be possible to pause and play the video. When participants decide to continue to the next page a notification will pop up, telling them they cannot return to this page when they continue. This functionality will be implemented so that participants cannot keep going back to the video to get help during their analyses and/or comparisons. This way participants are relying more on the feedback they receive and this way it is expected that the feedback types have a bigger influence. The video screen and its pop-up are shown in figure 3.7.

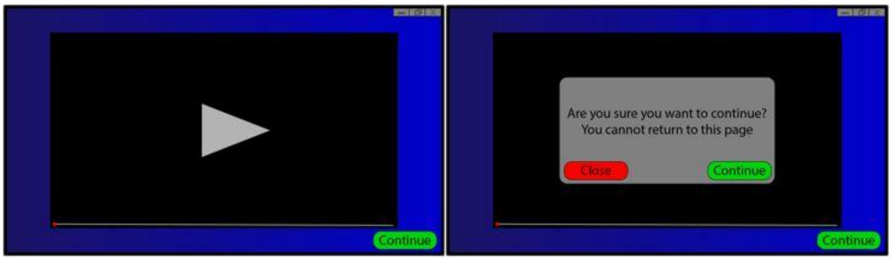

Are you sure you want to continue?
You cannot return to this page

Figure 3.7: Video screen and the pop-up

3.2.2 Analyses

When participants finished the introduction they will start on their first analysis. The analysis pages show a large fingerprint image on which participants can mark minutiae (figure 3.8). How to mark the minutiae and how to select the different labels and tolerances is summarized in the controls panel. Next to the fingerprint images the current selected labels and tolerances are visualized as well. This is done so that participants receive feedback from the application when one of the keys is pressed to change the selection. When participants click "continue"

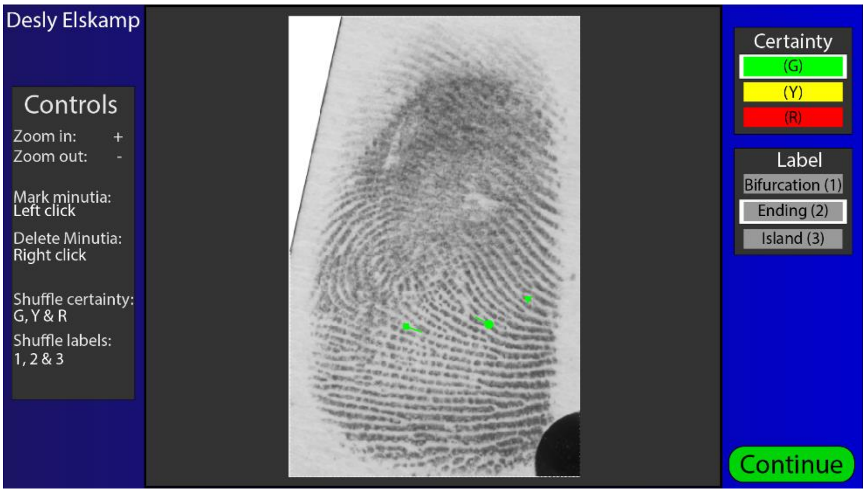


Figure 3.8: Analysis screen

to move on to the next fingerprint they will again get a pop-up telling them they cannot come back to this page. This functionality is essential to prevent any bias observations, which is caused by going back and forth between the two fingerprints.

During a discussion with the expert and supervisor a few changes were suggested. Firstly, a pattern comparison was recommended before starting the detailed analysis. Fingerprints are comprised of three main patterns and if those of the two fingerprints are not similar, a complete comparison would not be necessary, because the fingerprint would be excluded based on the overall pattern type. In the application this pattern comparison will therefore be implemented before starting the first analysis screen. Secondly, it was suggested to change the layout of the interface. Since people instinctively read a screen from left to right, the tolerance and label selections on the right caused confusion. Therefore, in the application the layout was changed, where the selections are displayed on the left side of the fingerprint. Lastly, it was suggested to add another label, called unknown.

3.2.3 Comparison

When participants finished the two analyses they will start the comparison. In the comparison phase the two analyses will be displayed side to side so that participants can go over all the minutiae they have marked during the analysis phase. When participants find more minutiae they should still be able to mark them, but this will automatically be done with an orange colour, following the GYRO standard [20]. Participants should also be able to scribble on the fingerprints and type notes in a text box. When the participants finished their comparison and made up their final conclusion, this conclusion can be entered by clicking the buttons as can be seen in figure 3.9.

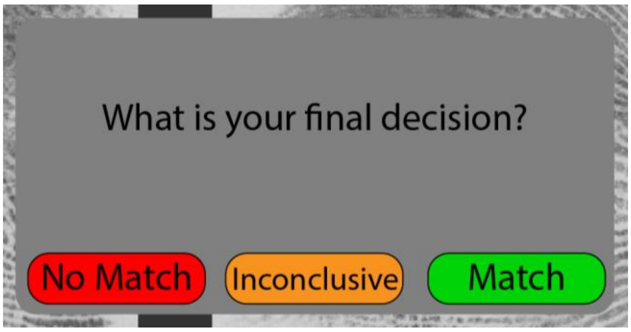

What is your final decision?

Figure 3.9: Final conclusion

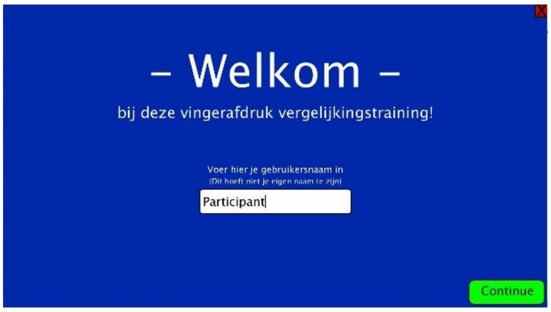
During a discussion with the expert and supervisor the way of entering the final conclusion was discussed. Eventually it was decided that a Likert scale would result in more interesting answers. Firstly, a Likert scale will show the certainty of the participant, since they are able to rate the fingerprints from ‘very likely to be from the same source’ to ‘very likely to be from different sources’. Secondly, there will be no neutral answer, as there will be six options. This way they participants are forced to make a decision heading to one of the two options.

4. The experiment

To gain more knowledge on which types of feedback are effective for developing fingerprint comparison skills an experiment has been conducted. This chapter will discuss the process of setting up the experiment from writing the application to how the data was processed. During this process weekly meetings with an expert and supervisor were organized to enhance the quality of the prototype and eventually the experiment. Before the final application was created a list of requirements was made to ensure all important aspects were implemented. The requirements are based on the literature findings, standards in the fingerprint comparison domain and discussions with the expert and supervisor. The list of requirements can be found in appendix A.

4.1 The application

The application for the experiment was created in Processing. This software was chosen because of the amount of experience of the researcher rather than functionality or reliability of the software and might therefore not necessarily be the most appreciated software by others. The entire code of one of the versions can be found in appendix B. The other versions, which will all be discussed in later paragraphs are derived from the version in appendix B. Since all the participants for this experiment are Dutch it was decided to use Dutch for the visible texts in the application.



– Welkom –

bij deze vingerafdruk vergelijkingstraining!

Voer hier je gebruikersnaam in
(Dit hoeft niet je eigen naam te zijn)

Participant

Figure 4.1: Welcome screen

When the application was opened by the participant it launched the welcome screen figure 4.1. On this screen the participant was asked to fill in a username. Only when the participant entered at least one symbol the "continue" button could be used to move on to the next page.

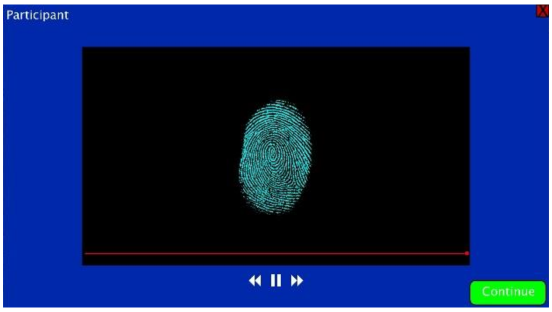


Figure 4.2: Video screen

On the next page (figure 4.2) the participant started by watching an explanatory video. In this video the participant was introduced to the fingerprint comparison domain and it explained them how a comparison should be done and how the application can be controlled. The video is in Dutch and the script can be found in appendix C. The participant could easily make jumps in the video by clicking on the timeline or using the fast forward or fast backward buttons. The participant could also pause and play the video and they were allowed to watch the video as many times as they wish for. However, once they continued to the next page the participant could not return to the video. This was done so that participants only got help from their assigned feedback type and not from the video. Only when the participant had reached the end of the explanatory video the "continue" button was enabled to continue to the next phase.

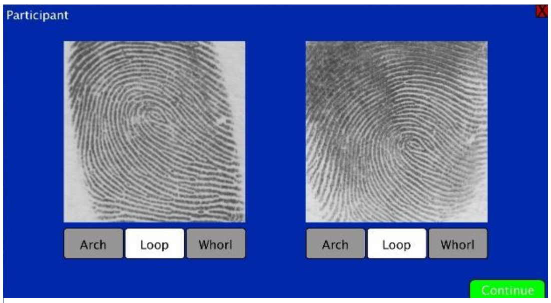
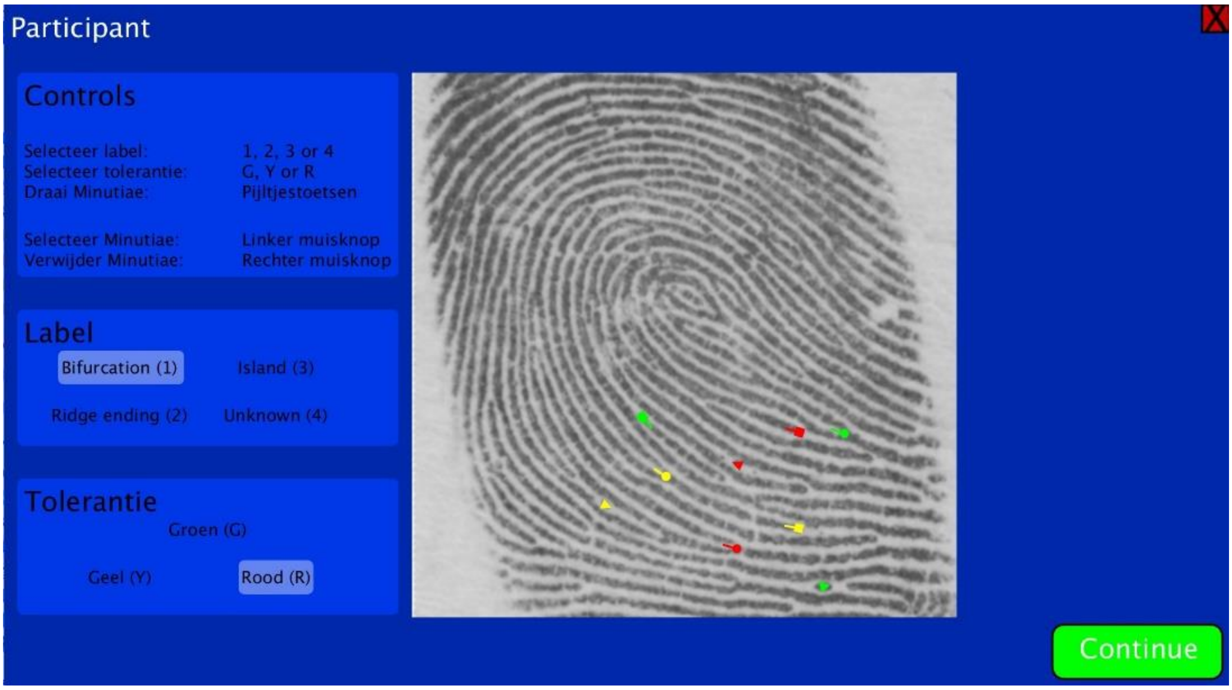


Figure 4.3: Pattern selection

In this next phase the participant were asked to select the patterns of both fingerprints. This phase was added to the application since a detailed analysis is not required when the patterns do not match. To select the right patterns the participant could simply click the buttons shown in figure 4.3 and then continue to the next phase.

In this phase the participant started their first analysis (figure 4.4). To mark minutiae in this latent print the participant could use their keys to scroll through the tolerances and labels. The arrow keys could be used to rotate the markers towards the right orientation. When the pointer of the participant was on top of the fingerprint image the standard pointer was replaced by the selected minutiae. By changing the selections on the screen and the shape, colour and orientation of the pointer, participants could directly see their actions visualized, which improves the user experience. Participants could use their left mouse button to mark the minutiae and their right mouse button to delete it. When the participant thought they found and marked all the minutiae they could continue to the next fingerprint. When this fingerprint was analysed as well the participant could continue to the comparison phase.


Participant

Controls

Selecteer label: 1, 2, 3 or 4
Selecteer tolerantie: G, Y or R
Draai Minutiae: Pijltjestoetsen

Selecteer Minutiae: Linker muisknop
Verwijder Minutiae: Rechter muisknop

Figure 4.4: Analysis screen

In the comparison phase the participant was still able to mark and delete minutiae. However, the minutiae always had an orange tolerance, so that it adheres to the GYRO standard [20]. Next to adjusting the minutiae participants could also use the scribble and note tools visualized in figure 4.5 and 4.6. To make drawings on the fingerprint the participant had to select the tool and use their left mouse button. When the participant wanted to erase a part of their drawings this could be done by using the right mouse button. Those tools could be utilised to clarify their way of thinking, which is important for documentation. A clear documentation will be easier to review by others and will lead to a more reliable testimony.

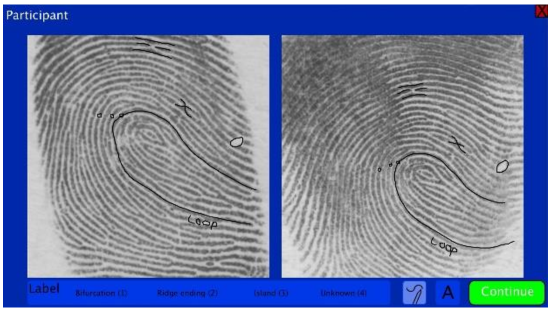


Figure 4.5: Scribbling tool

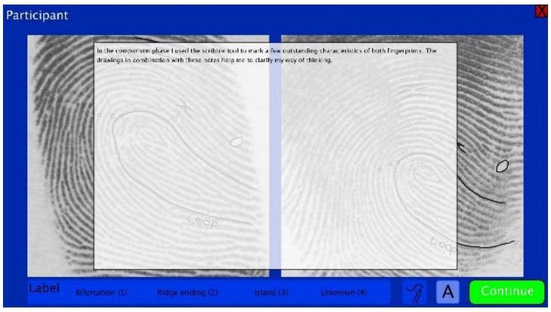

In the comparison phase I used the scribble tool to mark a few outstanding characteristics of both fingerprints. The drawings in combination with these notes help me to clarify my way of thinking.

Figure 4.6: Notes tool

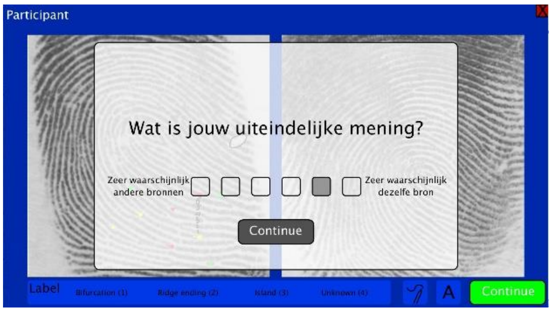

Wat is jouw uiteindelijke mening?

Figure 4.7: Saving the final opinion

When the participant finished the comparison phase, they were asked to save their final opinion. Do they think the fingerprints were taken from the same source and how reliable do they think their opinion is? To save their opinion the participant could simply select one of the squares of the Likert scale and continue to the next page (figure 4.7). The process of selecting the patterns, analysing the prints and comparing the analyses were repeated two more times. When the participant finished the last comparison they were thanked for their help and the application could be closed.

4.2 Implemented feedback

The previous paragraphs explain how the application worked for participants in the '*No feedback'* group. However, the goal of the research is to compare different feedback types, so these were implemented in the application as well. Therefore, the following paragraphs explain how the different types of feedback were presented to the participants.

4.2.1 Current Feedback

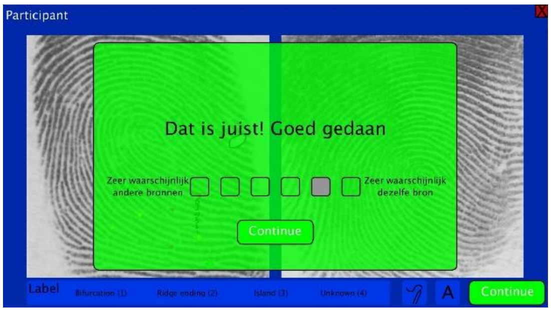

Dat is juist! Goed gedaan

Figure 4.8: Current feedback

In current practise analysts often just get feedback on their final decision. Therefore, the participants in the '*Current feedback'* group just got feedback on their final answers in the pattern selection and the overall comparison. As can be seen is figure 4.8, the participant got direct feedback confirming if their answer was correct or incorrect. If the given answer was correct the participant could continue to the next page. When the given answer was incorrect the participant was asked to review their decision and try again. This feedback type might seem very simple and lacking information, but that is exactly the problem that many analysts face during their training.

4.2.2 Task Feedback

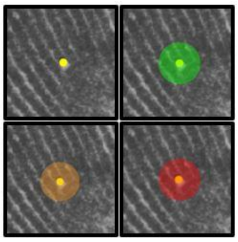


Figure 4.9: Task feedback

This feedback type complements the *'Current feedback'* type by providing direct feedback on all the minutiae marked by the participant. As can be seen in figure 4.9 the participant received the feedback by green, yellow and red flashes. These flashes indicated if the minutiae was marked correctly, semi-correctly or incorrectly. This direct feedback could hopefully solve misunderstanding relatively quickly, keeping the participant from making the same mistake over and over again.

4.2.3 Peer Feedback

Participants in the peer feedback group did not receive any direct feedback from the application. The participant first did their own comparison and thereafter had to critically review a comparison done by another person. To review the comparison the participant was provided with the marked minutiae, notes and scribbles. During the review the participant was able to mark minutiae with white markings and they could use the scribbling and notes tools as well. When this was done the participant was done with the first half and had to wait for their feedback. Just as in real-life scenarios, this could take up to several working days. When their work had been checked they were provided with the adjusted minutiae and extra drawings and notes. After studying their feedback, the participant was ready to finish their final comparison.

4.3 Survey

To gain more insight in the satisfaction of participants of the different feedback groups each participant was asked to fill in a quick survey after they finished the training. The survey was kept as short as possible, since the training itself already takes about one-and-a-half hour to finish. The questions of the survey can be found in appendix D. Those questions mainly focussed on how the participant experienced the feedback and if the participant thinks the feedback was effective. Eventually the answers of participants were compared to each other to see which feedback type leads to most satisfaction.

4.3 Participants

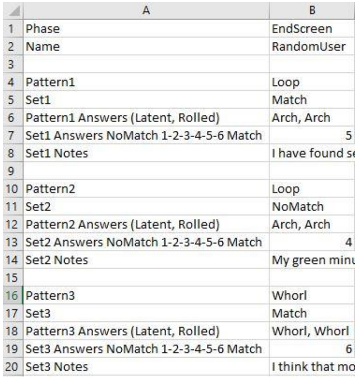
Social contacts of the researcher were approached to serve as participants. This is justifiable since participants did not have to review the work of the researcher, which makes any biased opinions irrelevant. The only requirements for participants were that they should be adults and did not have any prior experiences with fingerprint comparison or other relative perceptual expertise (for example: analysing blood spatters or facial recognition).

A total of fourteen people agreed to participate in the experiment and those people were randomly assigned to one of the feedback types. The distribution of the fourteen participants was as follows: three people were assigned to the *'No Feedback'* group, three to the *'Current practise'* group, four to the *'Task feedback'* group and the last four to the *'Peer feedback'* group. Before participants could start with the training they first had to fill in the consent form. The brochure and consent form, which are both in Dutch, can be found in appendix E. Eventually eleven participants completely finished the training. One participant of the *'No feedback'* group dropped out and two in the *'Task feedback'* group dropped out as well.

4.4 Processing data

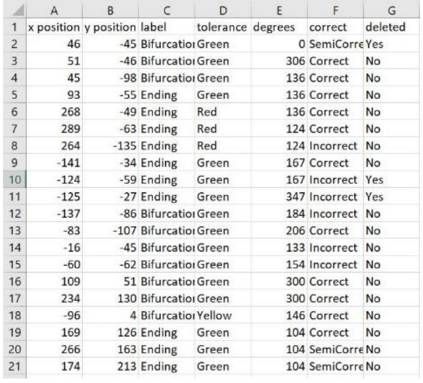
To eventually compare the results from the different feedback types the application automatically stored the necessary data on the computer of the participant. When the participant finished the training they were asked to send the data folder back to the researcher.

As can be seen in figure 4.10 and 4.11 the application saved the answers given at the pattern selection and the final opinion, all the marked minutiae, even if the marking was deleted by the participant and the notes in excel files. For the marked minutiae the application saved the coordinates, the label, the tolerance, the orientation, if the marking was correct and if the marking was deleted by the participant. To save the scribbles made by the participant a screenshot of the comparisons was saved as well.



	A	B
1	Phase	EndScreen
2	Name	RandomUser
3		
4	Pattern1	Loop
5	Set1	Match
6	Pattern1 Answers (Latent, Rolled)	Arch, Arch
7	Set1 Answers NoMatch 1-2-3-4-5-6 Match	5
8	Set1 Notes	I have found se
9		
10	Pattern2	Loop
11	Set2	NoMatch
12	Pattern2 Answers (Latent, Rolled)	Arch, Arch
13	Set2 Answers NoMatch 1-2-3-4-5-6 Match	4
14	Set2 Notes	My green minu
15		
16	Pattern3	Whorl
17	Set3	Match
18	Pattern3 Answers (Latent, Rolled)	Whorl, Whorl
19	Set3 Answers NoMatch 1-2-3-4-5-6 Match	6
20	Set3 Notes	I think that mo

Figure 4.10: Stats saved by the application



	A	B	C	D	E	F	G
1	x position	y position	label	tolerance	degrees	correct	deleted
2	46	-45	Bifurcatio	Green	0	SemiCorre	Yes
3	51	-46	Bifurcatio	Green	306	Correct	No
4	45	-98	Bifurcatio	Green	136	Correct	No
5	93	-55	Ending	Green	136	Correct	No
6	268	-49	Ending	Red	136	Correct	No
7	289	-63	Ending	Red	124	Correct	No
8	264	-135	Ending	Red	124	Incorrect	No
9	-141	-34	Ending	Green	167	Correct	No
10	-124	-59	Ending	Green	167	Incorrect	Yes
11	-125	-27	Ending	Green	347	Incorrect	Yes
12	-137	-86	Bifurcatio	Green	184	Incorrect	No
13	-83	-107	Bifurcatio	Green	206	Correct	No
14	-16	-45	Bifurcatio	Green	133	Incorrect	No
15	-60	-62	Bifurcatio	Green	154	Incorrect	No
16	109	51	Bifurcatio	Green	300	Correct	No
17	234	130	Bifurcatio	Green	300	Correct	No
18	-96	4	Bifurcatio	Yellow	146	Correct	No
19	169	126	Ending	Green	104	Correct	No
20	266	163	Ending	Green	104	SemiCorre	No
21	174	213	Ending	Green	104	SemiCorre	No

Figure 4.11: Minutiae saved by the application

With the use of all this data an analysis was done to compare the answers and the behaviour of participants in the different feedback groups. This analysis did not only focus on the quality of the given answers, but also on the number of marked minutiae, if the participant was able to recover mistakes and how confident the participant was about their own work. If any other outstanding things were discovered this is discussed in the results section as well.

5. Experiment results

This section will thoroughly discuss the results from the experiment. First an overview of the overall scores will be given and thereafter the remarkable results from each feedback type will be discussed. It is important to realize that these results are based on a limited number of participants and comparisons and are therefore not reliable enough to draw conclusions with high confidence. Another limitation is that not all the minutiae are checked by hand, but by the application. The application uses a certain margin for positioning the minutiae and if the participant positioned the minutiae too far off, this minutiae is labelled as incorrect even though it could be considered as semi-correct. The overall results do not show significant differences between the different feedback types.

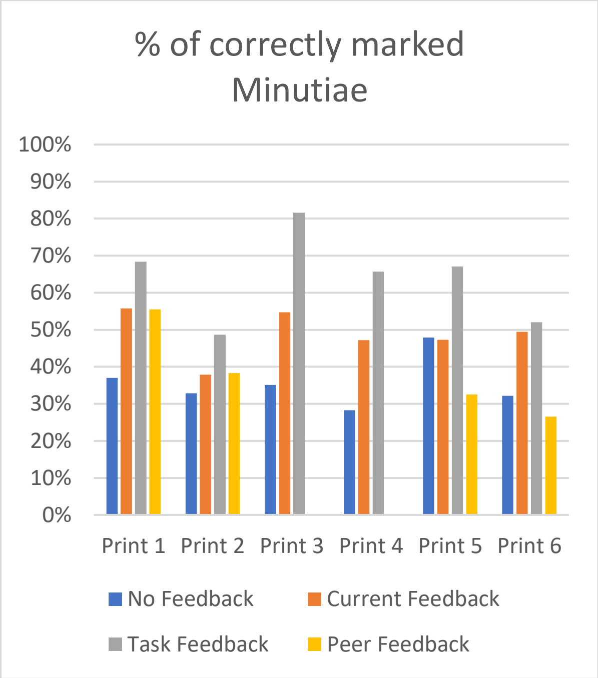


Figure 5.1: Percentage of correctly marked Minutiae

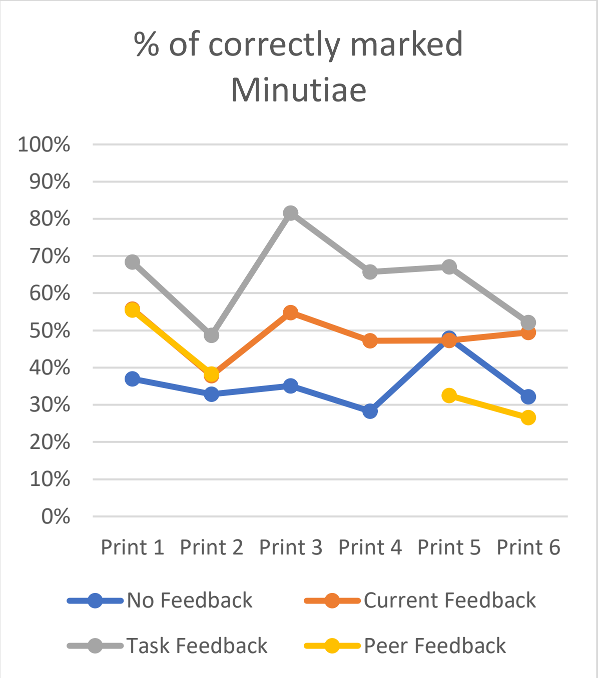


Figure 5.2: Percentage of correctly marked Minutiae as a trend

In figures 5.1 and 5.2 the percentage of correctly marked minutiae is visualized for each feedback type. The graphs do not show any information for the '*peer feedback'* group for the third and fourth comparison, because the participants in this group had to give feedback instead of analysing the fingerprints themselves. The figures do show differences between the groups, but it seems like these are caused by the skill level of the participants rather than the impact of the feedback. The trendline of the different groups do not show a clear relationship between results and feedback either. The difference in performances over the different prints seems to be caused by the difficulty of the prints, rather than the development of skills. Worse results in the end might also indicate a decrease of concentration after working on the experiment for over an hour.

The final opinions of participants do not show a significant difference either. The data shows that all participants gave the right answer after the second and third comparison, some with a higher confidence than others. However, it is interesting to see that four participants gave the incorrect answer at the first comparison, so either there is some progress in understanding fingerprint comparisons, or the first comparison was simply more difficult than the other two.

To get more insight in the user experience amongst the different groups the results of the survey were analysed as well. Participants in all groups, except for the '*No feedback'* group, described the feedback as useful and stated their certainty had increased during the training. One outstanding outcome of the survey is that both participants in the '*Task feedback'* group marked the feedback as slightly annoying.

When the participants were asked which sort of feedback they would like to see added most participants mentioned a kind of feedback that gives some sort of confirmation. Participants in the '*No feedback'* group asked for feedback similar to what participants in the '*Current feedback'* group received and the participants asked for feedback similar to what participants in the '*Task feedback'* group received. This shows that people often look for some sort of confirmation to increase their self-confidence.

5.1 No Feedback & Current Feedback

In these two groups participants made some recurring mistakes. Examples of these are: marking relatively short lines as islands rather than two ridge endings, marking very small interruptions in ridges as ridge endings and marking ridges at the side of the print that immediately stop as islands, while it is more likely those are just ridge endings. Because of the lack of feedback in both situations, these mistakes were made in all three fingerprints. Even though their final opinions were correct, these kinds of mistakes can lead to problems in the documentation and more complex comparisons. By providing some very simple feedback these problems could have been solved, so that these participants do not make the same mistake over and over again.

5.2 Task Feedback

In the '*Task feedback'* group two completely different behaviours can be found in the data. The first participant made quite some typical mistakes similar to the ones explained in the previous paragraph. However, because of the direct task feedback most of these incorrect markings were deleted and replaced by new ones. When the system showed this participant a red flash they reviewed their decision, but they did not always delete it. This shows that this participant used the feedback to get confirmation, but still trusted their own skills. The second participant made the same typical mistakes at the first print, but rather than reviewing their decision it seems like they started guessing. Even though some mistakes were very clear, like using an incorrect label, they placed and deleted the same type of minutiae over and over again on slightly different coordinates. This show that this participant focussed more on getting the right answer rather than understanding the process, which is a pitfall of providing task feedback.

5.3 Peer Feedback

As in the other groups the participants in the '*Peer feedback'* group made the same typical mistakes next to some other small misunderstandings. After receiving the feedback most of these typical issues were solved, but some of the feedback also led to new problems. For example, one of the participants marked a relatively small number of minutiae and the feedback told them to take more time to find more minutiae to mark. In the final comparison they tried to find more minutiae, but this led to a lot of incorrect markings, like marking very small interruptions as ridge endings. Other participants also showed improvement with some misunderstandings but did still not mark the minutiae perfectly. This shows that the main fallback of using peer feedback is that all the feedback is combined in one message and not given directly after the actions of participants.

6. Discussion

In this section the research questions as stated in the introduction will be answered. The literature findings and results from the experiment will be discussed in the conclusion. Thereafter, limitations and recommendations for future work will be discussed as well.

6.1 Conclusion

During the literature research a few promising study methods were found relevant to training perceptual expertise. Feedback is the most often mentioned method, but most of the studies focus on the development of overall perceptual expertise and do not focus on fingerprint comparison. Therefore, the focus of this research was on finding the most effective feedback types for training novices the basics of fingerprint comparison.

Although the experiment was small scaled, the different feedback groups show some interesting results. The participants in the '*No feedback'* and '*Current feedback'* groups showed the same typical mistakes in all three comparisons. Meaning that even though the final opinion might be correct, their opinion can be considered as less reliable. The direct feedback that participants in the '*Task feedback'* group received shows that simple feedback could have solved the typical misunderstandings very quickly, which keeps the participants from making the same mistake over and over again. The main drawback of this task feedback is that participants might focus more on getting the right answer, rather than understanding the process of fingerprint comparison. Finally, the '*Peer feedback'* group shows that personalized feedback can be very useful and can solve misunderstanding by the participants. However, peer feedback is not received directly and this causes the feedback to be lagging behind. Where participants in the *'Task feedback'* group were able to immediately review their decisions when they were incorrect, participants in the '*Peer feedback'* group had to wait up to several days to get all their feedback at once. In a future e-learning environment this would mean that the users will have to wait for feedback after every comparison, or they will make unnecessary mistakes during the comparisons they decide to do before receiving the feedback.

Other methods found during the literature research focussed on making users more aware of details while comparing images. These methods were not implemented in the experiment, since the focus was on the feedback types. However, the results of the experiment do show the importance of these methods, because participants indeed used partial matching in some of the cases. When participants found a few similarities they often formed their opinion without critically going by all the minutiae they had marked. In future studies the impact of forcing participants to study details could be very interesting to test.

It can be concluded that improving the educational system for (future) fingerprint experts with the use of researched study methods and e-learning still requires a lot of further research. This paper discussed several aspects that should be considered and it may be seen as a first step in the process. Some feedback types show some interesting and promising characteristics, but its significance will still have to be proven. The recommended future work can be found in later paragraphs.

6.2 Limitations

Firstly, during this research social contact was discouraged because of Covid-19 regulations and all the meetings were held online. This did not have a large impact on the overall research, since e-learning can be done remotely anyways, but real-life meetings could have made the ideation phase easier and more valuable. Normally the first step in the ideation phase would have been to create paper prototypes and test them with several people. However, in this situation the paper prototypes were replaced by online versions created on adobe illustrator. This caused that playtesting could not be done by the participants themselves but was controlled by the researcher. Participants could explain which action they wanted to do, but it would be the researcher clicking the buttons. If those playtests would have been done in real life more flaws would possibly have been found.

Secondly, the lack of participants caused unreliable results. The experiment took about one-and-a-half hour for each participant to complete and it is hard to find people who are willing to spend such a large amount of time on this research. Eventually enough participants were found to get some more insight in the different feedback types, but for more reliable results the different groups should be much larger. To achieve this, participants should probably be rewarded for the amount of work they have to do. When more participants are found it will also be possible to test more feedback types.

6.3 Future work

This research is the first step in creating an e-learning environment for fingerprint analysts. However, a lot of research will still have to be done to explore all the missing pieces. Firstly, the experiment that has been conducted should be upscaled to gather more reliable results. Redoing this experiment with more participants and with all the participants doing more comparisons will result in more informative trend lines and possible differences between the feedback groups. Afterwards, the results should be analysed by people more familiar with the fingerprint comparison domain, so that they those analyses are more reliable and more remarkable characteristics are found.

Secondly, there are many pieces of the puzzle that are still missing. More feedback types should be tested and also other study methods that were found during the literature research should be considered. The different feedback types and researched study methods can also lead to different results in different situations. Further research should thus show out which methods lead to significant improvements in which phases of developing fingerprint comparison skills. Eventually, more research should be done on the topic of e-learning as well. This research discusses the basics, but more literature research and experiments should show the effectiveness of the software tool.

Thus, this research could be a first step in designing a new e-learning environment for fingerprint analysts, but further research should be conducted to explore all the missing pieces.

References

[1] H. Lee and R. Gaensslen, *Advances in fingerprint technology*. Boca Raton, Fla.: CRC Press, 2001.

[2] *Fingerprint Analyst: Certification & Training Requirements,* 2020 December [online]. Available: https://study.com/articles/Fingerprint_Analyst_Training_Requirements_and_Career_Info.html#:~:text=To%20become%20a%20fingerprint%20analyst,%2C%20biology%2C%20and%20criminal%20justice

[3] "How to Become a Fingerprint Analyst: 2020 Guide - CSI Accelerator", *CSI Accelerator*, 2021. [Online]. Available: https://csiaccelerator.org/careers/fingerprint-analyst/.

[4] R. Searston and J. Tangen, "Training perceptual experts: Feedback, labels, and contrasts.", *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, vol. 71, no. 1, pp. 32-39, 2017.

[5] D. White, R. Kemp, R. Jenkins and A. Burton, "Feedback training for facial image comparison", *Psychonomic Bulletin & Review*, vol. 21, no. 1, pp. 100-106, 2013.

[6] M. Herzog and M. Fahle, "The role of feedback in learning a vernier discrimination task", *Vision Research*, vol. 37, no. 15, pp. 2133-2141, 1997.

[7] A. Petrov, B. Dosher and Z. Lu, "Perceptual learning without feedback in non-stationary contexts: Data and model", *Vision Research*, vol. 46, no. 19, pp. 3177-3197, 2006.

[8] V. Popov and L. Reder, "Greater discrimination difficulty during perceptual learning leads to stronger and more distinct representations", *Psychonomic Bulletin & Review*, vol. 27, no. 4, pp. 768-775, 2020.

[9] J. Hattie And H. Timperley, “The Power of Feedback”, Review of Educational Research, vol. 77, no. 1, pp. 81-112, 2007.

[10] M. Curtis, C. Maraj, M. Ritman and F. Jentsch, "Investigation of the Impact of Feedback on Decision Accuracy and Reaction Time in a Perceptual Training Task", *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 54, no. 19, pp. 1630-1634, 2010.

[11] V. Arkorful and N. Abaidoo, “The Role of e-Learning, the Advantages and Disadvantages of Its Adoption in Higher Education”, International Journal of Education and Research, no. 2, pp. 397-410, 2014

[12] A. Biedermann et al., "E-learning initiatives in forensic interpretation: Report on experiences from current projects and outlook", *Forensic Science International*, vol. 230, no. 1-3, pp. 2-7, 2013.

[13] C. Liu, “The comparison of learning effectiveness between traditional face-to-face learning and e-learning among goal-oriented users” 6th International Conference on Digital Content, Multimedia Technology and its Applications, Seoul, Korea (South), 2010, pp. 255-260

[14] I. Koksal, "The Rise Of Online Learning", *Forbes*, 2021. [Online]. Available: https://www.forbes.com/sites/ilkerkoksal/2020/05/02/the-rise-of-online-learning/?sh=772e5c9272f3.

[15] A. Domínguez, J. Saenz-de-Navarrete, L. de-Marcos, L. Fernández-Sanz, C. Pagés and J. Martínez-Herráiz, "Gamifying learning experiences: Practical implications and outcomes", *Computers & Education*, vol. 63, pp. 380-392, 2013.

[16] W. Chapman and R. Hicklin, ACEware latent fingerprint identification research and software development.

[17] "PiAnoS documentation — PiAnoS - Picture Annotation System dev documentation", Ips-labs.unil.ch. [Online]. Available: https://ips-labs.unil.ch/doc/index.html.

[18] "Automated Fingerprint Identification System (AFIS) overview - A short history.", *Thales Group*, 2021. [Online]. Available: https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/biometrics/afis-history.

[19] B. Bonnaz, M. et al., "Automatic assessment of fingermarks quality: exploration of the possible application in the context of detection and comparison with human examiners", Journal of Forensic Sciences, 2021.

[20] Langenburg, G. & Champod, Christophe, “The GYRO system - a recommended approach to more transparent documentation”. Journal of Forensic Identification, vol. 61, pp. 373-384, 2011.

Appendices

Appendix A

The initial list of requirements for the prototype

Introduction

In this application, users need to be able to complete fingerprint analyses and comparisons. Because we want to study the learning curve of users, they should not be able to repeat any exercises, or watch back the explanatory video. The following paragraphs explain all the functionalities that should be in the application, complemented by functionalities that are useful, but not a 100% necessary.

Essential functionalities

- Users should be able to mark minutiae (location, orientation, label and certainty) on top of fingerprint images.
- Users should be able to delete marked minutia so that they can recover mistakes or mis clicks
- Users should be able to continue their work if they had a break (also if they closed the application)
    - To allow this, the data should be stored in a separate (excel) document that will be used by the application
- In the comparison phase (and giving feedback phase with ‘peer feedback method’), users should be able to draw on the analyses and/or write down comments in a text box.
- The application should track the following stats:
    - Number of correctly marked minutiae and their confidence rate
    - Number of ‘almost’ correctly marked minutiae (correct position, but incorrect label or orientation)
    - Number of incorrectly marked minutiae
    - If the final conclusion was correct
- Users should not be able to re-do any analyses or re-watch the video when they continued to the next page (video can be watched multiple times, but when they moved to the next page, they can’t go back)

Functionalities that are useful, but not essential

- Users can fill in the data that we need in the same application to simplify their process (implemented survey)
- The explanatory videos are implemented in the application (if not, they will be provided next to the application)
- Users are able to adjust marked minutiae without the need of completely deleting it (more difficult to implement, but could enhance the user experience)
- Users are able to zoom in on fingerprints

Appendix B

The code of the ‘Task Feedback’ version in Processing 3.5.4.

```
import processing.video.*;
Table Stats;
String Phase;
PApplet app;

WelcomeScreen welcomeScreen;
VideoScreen videoScreen;
PatternScreen patternScreen;
AnalysisScreen analysisScreen;
ComparisonScreen comparisonScreen;
MarkMinutiae markMinutiae;
DrawMinutiae drawMinutiae;
DeleteMinutiae deleteMinutiae;
CheckMinutiae checkMinutiae;
FinalScreens finalScreens;
ArrayList<DirectFeedback> directFeedback;


void setup() {
  //Setting up the screen
  fullScreen();
  surface.setTitle("Fingerprint comparison");
  surface.setLocation((width-1800)/2, (height-900)/2 - 80);
  surface.setSize(1800, 1000);
  app = this;

  welcomeScreen = new WelcomeScreen();
  videoScreen = new VideoScreen();
  patternScreen = new PatternScreen();
  analysisScreen = new AnalysisScreen();
  comparisonScreen = new ComparisonScreen();
  markMinutiae = new MarkMinutiae();
  drawMinutiae = new DrawMinutiae();
  deleteMinutiae = new DeleteMinutiae();
  checkMinutiae = new CheckMinutiae();
  directFeedback = new ArrayList<DirectFeedback>();
  finalScreens = new FinalScreens();
}

void draw() {
  //Get the current Phase from table
  Stats = loadTable("Stats.csv");
  TableRow row = Stats.getRow(0);
  Phase = row.getString(1);

  background(0, 40, 170);

  //Close Program button
  fill(200, 0, 0);
  stroke(0);
  rectMode(CORNER);
  rect(width - 40, 0, 40, 40);
  fill(0);
  textSize(60);
  textAlign(LEFT);
  text("X", width - 37, 40);

  //Run methods acoording to the Phase
  //Welcome phase
  if (Phase.equals("WelcomeScreen")) {
    welcomeScreen.Run();
  }
  //Video phase
  if (Phase.equals("VideoScreen")) {
    videoScreen.Run();
  }
  //Pattern phases
  if (Phase.equals("PatternScreen1") || Phase.equals("PatternScreen2") ||
Phase.equals("PatternScreen3")) {
    patternScreen.Run(Phase);
  }
  //Analysis phases
  if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
    analysisScreen.Display(Phase);
    //show feedback flashes
    if (Phase.equals("LatentPrint3") == false && Phase.equals("RolledPrint3") == false &&
Phase.equals("Comparison3") == false) {
      for (int i = directFeedback.size()-1; i>=0; i--) {
        DirectFeedback feedback = directFeedback.get(i);
        feedback.Display();
        //remove flash after some time
        if (feedback.isDead()) {
          directFeedback.remove(i);
        }
      }
    }
    drawMinutiae.Run(Phase);
    if (analysisScreen.onPrint(mouseX, mouseY)) {
      markMinutiae.Mouse(Phase, mouseX, mouseY);
      noCursor();
    } else {
      cursor();
    }
  }
  //Comparison phases
  if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
    comparisonScreen.Run(Phase, mouseX, mouseY);
    if (Phase.equals("Comparison3") == false) {
      for (int i = directFeedback.size()-1; i>=0; i--) {
        DirectFeedback feedback = directFeedback.get(i);
        feedback.Display();
        //remove flash after some time
        if (feedback.isDead()) {
          directFeedback.remove(i);
        }
      }
    }
    drawMinutiae.Run(Phase);
    if (comparisonScreen.onLatentPrint(mouseX, mouseY) ||
comparisonScreen.onRolledPrint(mouseX, mouseY)) {
      markMinutiae.Mouse(Phase, mouseX, mouseY);
      noCursor();
    } else {
      cursor();
    }
    comparisonScreen.Popup();
    comparisonScreen.Decision();
    comparisonScreen.ScreenShot();
  }
  //Screen which introduces the final set
  if (Phase.equals("StartFinal")) {
    finalScreens.StartFinal();
  }
  //Final screen
  if (Phase.equals("EndScreen")) {
    finalScreens.End();
  }
}


void keyPressed() {
  if (keyCode != ESC) {
    //Allow typing in welcome phase
    if (Phase.equals("WelcomeScreen")) {
      if ((key >= &apos;A&apos; && key <= &apos;Z&apos;) || (key>= &apos;a&apos; && key <=
&apos;z&apos;) || key == &apos; &apos;) {
        welcomeScreen.Typing(key);
      } else if (keyCode == BACKSPACE) {
        welcomeScreen.BackSpace();
      }
    }
    //Allow selecting tolerance, label and degrees in analysis phases and comparison phases
    if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3") ||
      ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3"))&& comparisonScreen.Function == "Minutiae")) {
      if (keyCode == RIGHT) {
        markMinutiae.Select(&apos;.&apos;);
      } else if (keyCode == LEFT) {
        markMinutiae.Select(&apos;,&apos;);
      } else {
        markMinutiae.Select(key);
      }
    }
    //Allow users to type in the comparisons Notes phase
    if ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) && comparisonScreen.Function == "Notes") {
      comparisonScreen.KeyPressed(key);
    }
  } else {
    //Cannot close program with escape
    key = 0;
  }
}

void keyReleased() {
  //Stop the rotation of minutiae in analysis phases and comparison phases
  if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("Comparison1") || Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") ||
Phase.equals("Comparison2") ||
    Phase.equals("LatentPrint3") || Phase.equals("RolledPrint3") ||
Phase.equals("Comparison3")) {
    markMinutiae.KeyReleased();
  }
}

void mouseClicked() {
  //Close program
  if (mouseY > 0 && mouseY < 40 && mouseX > width-40 && mouseX < width) {
    exit();
  }
  //Mouseclicked combined with its phase calls different methods
  if (Phase.equals("WelcomeScreen")) {
    welcomeScreen.MouseClicked(mouseX, mouseY);
  }

  if (Phase.equals("VideoScreen")) {
    videoScreen.MouseClicked(mouseX, mouseY);
  }

  if (Phase.equals("PatternScreen1") || Phase.equals("PatternScreen2") ||
Phase.equals("PatternScreen3")) {
    patternScreen.MouseClicked(Phase, mouseX, mouseY);
  }

  if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
    analysisScreen.MouseClicked(Phase, mouseX, mouseY);
    if (analysisScreen.onPrint(mouseX, mouseY)) {
      if (mouseButton == LEFT) {
        markMinutiae.MouseClickedLeft(Phase, mouseX, mouseY);
      }
      if (mouseButton == RIGHT) {
        deleteMinutiae.Delete(Phase, mouseX, mouseY);
      }
    }
  }

  if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
    comparisonScreen.MouseClicked(Phase, mouseX, mouseY);
    if (comparisonScreen.onLatentPrint(mouseX, mouseY) ||
comparisonScreen.onRolledPrint(mouseX, mouseY)) {
      if (mouseButton == LEFT) {
        markMinutiae.MouseClickedLeft(Phase, mouseX, mouseY);
      }
      if (mouseButton == RIGHT) {
        deleteMinutiae.Delete(Phase, mouseX, mouseY);
      }
    }
  }

  if (Phase.equals("StartFinal") || Phase.equals("EndScreen")){
    finalScreens.MouseClicked(Phase,  mouseX, mouseY);
  }
}

void mouseReleased() {
  //Mousereleased will use ScribbleReleased to break lines
  if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
    comparisonScreen.ScribbleReleased(mouseX, mouseY);
  }
}
```

```
class AnalysisScreen {
  Table Stats;
  String Phase;
  PImage LatentPrint1 = loadImage("LatentPrint1.png");
  PImage RolledPrint1 = loadImage("RolledPrint1.png");
  PImage LatentPrint2 = loadImage("LatentPrint2.png");
  PImage RolledPrint2 = loadImage("RolledPrint2.png");
  PImage LatentPrint3 = loadImage("LatentPrint3.png");
  PImage RolledPrint3 = loadImage("RolledPrint3.png");
  //Position and size of images
  float xpos = 1000;
  float ypos = height/2;
  float size = 800;
  boolean Continued = false;
  String username;

  AnalysisScreen() {
  }

  void Display(String p) {
    Phase = p;
    //Get username to display
    Stats = loadTable("Stats.csv");
    TableRow row = Stats.getRow(1);
    username = row.getString(1);
    //Display the right fingerprint image
    imageMode(CENTER);
    if (Phase.equals("LatentPrint1")) {
      image(LatentPrint1, xpos, ypos, size, size);
    }
    if (Phase.equals("RolledPrint1")) {
      image(RolledPrint1, xpos, ypos, size, size);
    }
    if (Phase.equals("LatentPrint2")) {
      image(LatentPrint2, xpos, ypos, size, size);
    }
    if (Phase.equals("RolledPrint2")) {
      image(RolledPrint2, xpos, ypos, size, size);
    }
    if (Phase.equals("LatentPrint3")) {
      image(LatentPrint3, xpos, ypos, size, size);
    }
    if (Phase.equals("RolledPrint3")) {
      image(RolledPrint3, xpos, ypos, size, size);
    }
    //Show username and run methods
    textAlign(LEFT, TOP);
    textSize(40);
    fill(255);
    text(username, 10, 10);
    controlsDisplay();
    labelDisplay();
    toleranceDisplay();

    //Continue button
    textAlign(LEFT);
    textSize(40);
    stroke(0);
    rectMode(CENTER);
    fill(0, 255, 0);
    rect(1665, 950, 250, 80, 20);
    fill(255);
    text("Continue", 1580, 960);

    //Warning and cancel button
    if (Continued) {
      fill(200);
      rect(1600, 800, 380, 200, 20);
      fill(0);
      textSize(20);
      text("Weet je zeker dat je wilt doorgaan?", 1420, 750);
      text("Je kan niet terug naar deze pagina", 1420, 785);
      fill(255, 0, 0);
      rect(1665, 860, 250, 80, 20);
      fill(255);
      textSize(40);
      text("Cancel", 1600, 870);
    }
  }


  boolean onPrint(float x, float y) {
    //Check if the mouse is on top of the print
    if (x >= xpos - size/2 && x <= xpos + size/2) {
      if (y >= ypos - size/2 && y <= ypos + size/2) {
        return true;
      } else {
        return false;
      }
    } else {
      return false;
    }
  }

  void controlsDisplay() {
    //Show controls on a rect background
    rectMode(CORNER);
    noStroke();
    fill(0, 55, 230);
    rect(20, 100, 560, 300, 10);
    fill(0);
    textSize(40);
    textAlign(LEFT, TOP);
    text("Controls", 30, 110);
    textSize(25);
    text("Selecteer label:", 30, 200);
    text("Selecteer tolerantie:", 30, 230);
    text("Draai Minutiae:", 30, 260);
    text("Selecteer Minutiae:", 30, 330);
    text("Verwijder Minutiae:", 30, 360);

    text("1, 2, 3 or 4", 350, 200);
    text("G, Y or R", 350, 230);
    text("Pijltjestoetsen", 350, 260);
    text("Linker muisknop", 350, 330);
    text("Rechter muisknop", 350, 360);
  }

  void labelDisplay() {
    //Background rect
    fill(0, 55, 230);
    rect(20, 448, 560, 200, 10);
    //Selected rect
    fill(255, 100);
    if (markMinutiae.label.equals("Bifurcation")) {
      rect(80, 508, 185, 50, 10);
    }
    if (markMinutiae.label.equals("Ending")) {
      rect(65, 578, 210, 50, 10);
    }
    if (markMinutiae.label.equals("Island")) {
      rect(340, 508, 120, 50, 10);
    }
    if (markMinutiae.label.equals("Unclear")) {
      rect(320, 578, 165, 50, 10);
    }
    //text
    fill(0);
    textSize(40);
    text("Label", 30, 458);
    textSize(25);
    textAlign(CENTER, TOP);
    text("Bifurcation (1)", 170, 518);
    text("Ridge ending (2)", 170, 588);
    text("Island (3)", 400, 518);
    text("Unknown (4)", 400, 588);
  }

  void toleranceDisplay() {
    //Background rect
    fill(0, 55, 230);
    rect(20, 696, 560, 200, 10);
    //Selected rect
    fill(255, 100);
    if (markMinutiae.tolerance.equals("Green")) {
      rect(230, 746, 140, 50, 10);
    }
    if (markMinutiae.tolerance.equals("Yellow")) {
      rect(95, 816, 155, 50, 10);
    }
    if (markMinutiae.tolerance.equals("Red")) {
      rect(345, 816, 110, 50, 10);
    }
    //text
    fill(0);
    textAlign(LEFT, TOP);
    textSize(40);
    text("Tolerantie", 30, 706);
    textSize(25);
    textAlign(CENTER, TOP);
    text("Groen (G)", 300, 756);
    text("Geel (Y)", 170, 826);
    text("Rood (R)", 400, 826);
  }

  void MouseClicked(String p, float x, float y) {
    Phase = p;
    //If the continue button has been clicked go to next Phase
    if (x < 1790 && x > 1540 && y > 910 && y < 990) {
      if (Continued) {
        TableRow row = Stats.getRow(0);
        //Move to the next phase
        if (Phase.equals("LatentPrint1")) {
          row.setString(1, "RolledPrint1");
        }
        if (Phase.equals("RolledPrint1")) {
          row.setString(1, "Comparison1");
        }
        if (Phase.equals("LatentPrint2")) {
          row.setString(1, "RolledPrint2");
        }
        if (Phase.equals("RolledPrint2")) {
          row.setString(1, "Comparison2");
        }
        if (Phase.equals("LatentPrint3")) {
          row.setString(1, "RolledPrint3");
        }
        if (Phase.equals("RolledPrint3")) {
          row.setString(1, "Comparison3");
        }
        saveTable(Stats, "data/Stats.csv");
        Continued = false;
      } else {
        Continued = true;
      }
    }
    //If cancel button is clicked
    if (x < 1790 && x > 1540 && y > 820 && y < 900) {
      Continued = false;
    }
  }
}
```

```
class CheckMinutiae {
  SaveMinutiae saveMinutiae;
  Table Answers;
  String Phase;
  boolean answerCorrect;
  boolean posCorrect;
  float posMargin = 7;
  float degreesMargin = 12;

  float answerX;
  float answerY;
  float degrees;
  String answerL;
  Float answerD;

  CheckMinutiae() {
    saveMinutiae = new SaveMinutiae();
  }

  void Check(String p, float x, float y, String l, String t, float d) {
    Phase = p;
    //Negative degrees are translated to positive degrees
    if (d >= 0) {
      degrees = d;
    } else {
      degrees = d + 360;
    }

    //If in analysis phase, get the right answersheet
    if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
      if (Phase.equals("LatentPrint1")) {
        Answers = loadTable("Answers1.csv");
      }
      if (Phase.equals("RolledPrint1")) {
        Answers = loadTable("Answers2.csv");
      }
      if (Phase.equals("LatentPrint2")) {
        Answers = loadTable("Answers3.csv");
      }
      if (Phase.equals("RolledPrint2")) {
        Answers = loadTable("Answers4.csv");
      }
      if (Phase.equals("LatentPrint3")) {
        Answers = loadTable("Answers5.csv");
      }
      if (Phase.equals("RolledPrint3")) {
        Answers = loadTable("Answers6.csv");
      }
      //look through the answersheet to see if the given answer is (semi)correct
      for (int i = 1; i < Answers.getRowCount(); i++) {
        TableRow row = Answers.getRow(i);
        answerX = row.getFloat(0) + analysisScreen.xpos;
        answerY = row.getFloat(1) + analysisScreen.ypos;
        answerL = row.getString(2);
        answerD = row.getFloat(4);
        if (x > answerX - posMargin && x < answerX + posMargin) {
          if (y > answerY - posMargin && y < answerY + posMargin) {
            posCorrect = true;
            if ( l.equals(answerL)) {
              if ((abs(degrees - answerD) < degreesMargin && abs(answerD - degrees) <
degreesMargin) || l.equals("Island")) {
                answerCorrect = true;
                break;
              } else {
                answerCorrect = false;
              }
            } else {
              answerCorrect = false;
            }
            break;
          } else {
            answerCorrect = false;
            posCorrect =false;
          }
        } else {
          answerCorrect = false;
          posCorrect =false;
        }
      }
      saveMinutiae.Save(Phase, x, y, l, t, degrees, answerCorrect, posCorrect);
      directFeedback.add(new DirectFeedback(x, y, answerCorrect, posCorrect));
    }
    //If in comparison phase, get the right answersheets
    if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
      if (Phase.equals("Comparison1")) {
        if (comparisonScreen.onLatentPrint(x, y)) {
          Answers = loadTable("Answers1.csv");
        }
        if (comparisonScreen.onRolledPrint(x, y)) {
          Answers = loadTable("Answers2.csv");
        }
      }
      if (Phase.equals("Comparison2")) {
        if (comparisonScreen.onLatentPrint(x, y)) {
          Answers = loadTable("Answers3.csv");
        }
        if (comparisonScreen.onRolledPrint(x, y)) {
          Answers = loadTable("Answers4.csv");
        }
      }
      if (Phase.equals("Comparison3")) {
        if (comparisonScreen.onLatentPrint(x, y)) {
          Answers = loadTable("Answers5.csv");
        }
        if (comparisonScreen.onRolledPrint(x, y)) {
          Answers = loadTable("Answers6.csv");
        }
      }
      for (int i = 1; i < Answers.getRowCount(); i++) {
        TableRow row = Answers.getRow(i);
        if (comparisonScreen.onLatentPrint(x, y)) {
          answerX = row.getFloat(0) + comparisonScreen.xpos1;
        }
        if (comparisonScreen.onRolledPrint(x, y)) {
          answerX = row.getFloat(0) + comparisonScreen.xpos2;
        }
        answerY = row.getFloat(1) + comparisonScreen.ypos;

        answerL = row.getString(2);
        answerD = row.getFloat(4);
        if (x > answerX - posMargin && x < answerX + posMargin) {
          if (y > answerY - posMargin && y < answerY + posMargin) {
            posCorrect = true;
            if ( l.equals(answerL)) {
              if ( (abs(degrees - answerD) < degreesMargin && abs(answerD - degrees) <
degreesMargin) || l.equals("Island")) {
                answerCorrect = true;
                break;
              } else {
                answerCorrect = false;
              }
            } else {
              answerCorrect = false;
            }
            break;
          } else {
            answerCorrect = false;
            posCorrect =false;
          }
        } else {
          answerCorrect = false;
          posCorrect =false;
        }
      }
      saveMinutiae.Save(Phase, x, y, l, t, degrees, answerCorrect, posCorrect);
      directFeedback.add(new DirectFeedback(x, y, answerCorrect, posCorrect));
    }
  }

  void Display() {
    if (directFeedback.size() > 0) {
      for ( int i = directFeedback.size()-1; i >= 0; i--) {
        DirectFeedback feedback = directFeedback.get(i);
        feedback.Display();
      }
    }
  }
}
```

```
class ComparisonScreen {
  Table Stats;
  TableRow row;
  PImage LatentPrint1 = loadImage("LatentPrint1.png");
  PImage RolledPrint1 = loadImage("RolledPrint1.png");
  PImage LatentPrint2 = loadImage("LatentPrint2.png");
  PImage RolledPrint2 = loadImage("RolledPrint2.png");
  PImage LatentPrint3 = loadImage("LatentPrint3.png");
  PImage RolledPrint3 = loadImage("RolledPrint3.png");
  ArrayList<Scribble> scribble;

  String Function = "Minutiae";
  String Phase;
  String Username;
  String Notes = "";
  float xpos1 = width/2 - 420;
  float xpos2 = width/2 + 420;
  float ypos = height/2;
  float size = 800;
  int Opinion;
  int Guesses = 1;
  boolean AnswerGiven = false;
  boolean Correct;

  ComparisonScreen() {
    scribble = new ArrayList<Scribble>();
  }

  void Run(String p, float x, float y) {
    Phase = p;
    //get username to display it
    Stats = loadTable("Stats.csv");
    TableRow row = Stats.getRow(1);
    Username = row.getString(1);
    imageMode(CENTER);

    //display the right images
    if (Phase.equals("Comparison1")) {
      image(LatentPrint1, xpos1, ypos, size, size);
      image(RolledPrint1, xpos2, ypos, size, size);
    }
    if (Phase.equals("Comparison2")) {
      image(LatentPrint2, xpos1, ypos, size, size);
      image(RolledPrint2, xpos2, ypos, size, size);
    }
    if (Phase.equals("Comparison3")) {
      image(LatentPrint3, xpos1, ypos, size, size);
      image(RolledPrint3, xpos2, ypos, size, size);
    }
    textAlign(LEFT, TOP);
    textSize(40);
    fill(255);
    text(Username, 10, 10);
    labelDisplay();


    //Continue button
    textAlign(LEFT);
    textSize(40);
    stroke(0);
    rectMode(CENTER);
    fill(0, 255, 0);
    rect(1665, 950, 250, 80, 20);
    fill(255);
    text("Continue", 1580, 960);

    //Scribble button
    noStroke();
    fill(0, 55, 230);
    rect(1360, 950, 80, 80, 10);
    if (Function == "Scribble") {
      fill(255, 100);
      rect(1360, 950, 75, 75, 10);
    }
    noFill();
    stroke(0);
    arc(1347, 935, 25, 20, 0, PI);
    arc(1373, 935, 25, 20, PI, TWO_PI);
    beginShape();
    vertex(1365, 985);
    vertex(1380, 950);
    vertex(1380, 940);
    vertex(1370, 950);
    vertex(1355, 985);
    endShape(OPEN);

    //Notes button
    noStroke();
    fill(0, 55, 230);
    rect(1470, 950, 80, 80, 10);
    if (Function == "Notes") {
      fill(255, 100);
      rect(1470, 950, 75, 75, 10);
    }
    fill(0);
    textSize(60);
    text("A", 1450, 970);

    if (mousePressed && Function == "Scribble") {
      Scribble(x, y);
    }
    //Draw scribble
    stroke(0);
    for (int i = scribble.size()-1; i>0; i--) {
      Scribble singleScribble1 = scribble.get(i);
      Scribble singleScribble2 = scribble.get(i-1);
      if (singleScribble2.endpoint == false) {
        line(singleScribble1.posx, singleScribble1.posy, singleScribble2.posx, 
singleScribble2.posy);
      }
    }
  }

  void labelDisplay() {
    //Background rect
    fill(0, 55, 230);
    rectMode(CORNER);
    noStroke();
    rect(80, 910, 1200, 80, 10);
    //Selected rect
    if (Function == "Minutiae") {
      fill(255, 100);
      if (markMinutiae.label.equals("Bifurcation")) {
        rect(235, 925, 185, 50, 10);
      }
      if (markMinutiae.label.equals("Ending")) {
        rect(505, 925, 210, 50, 10);
      }
      if (markMinutiae.label.equals("Island")) {
        rect(825, 925, 120, 50, 10);
      }
      if (markMinutiae.label.equals("Unclear")) {
        rect(1045, 925, 165, 50, 10);
      }
    }
    //text
    fill(0);
    textSize(40);
    textAlign(LEFT);
    text("Label", 85, 950);

    textSize(25);
    text("Bifurcation (1)", 240, 960);
    text("Ridge ending (2)", 510, 960);
    text("Island (3)", 830, 960);
    text("Unknown (4)", 1050, 960);
  }

  void MouseClicked(String p, float x, float y) {
    Phase = p;
    if (y > 910 && y < 990) {
      //If continue button is clicked go to screenshot function
      if (x < 1790 && x > 1540) {
        if (Phase.equals("Comparison1")) {
          Function = "Screenshot1";
        }
        if (Phase.equals("Comparison2")) {
          Function = "Screenshot2";
        }
        if (Phase.equals("Comparison3")) {
          Function = "Screenshot3";
        }
      }
      //If label bar is clicked
      if (x > 80 && x < 1280) {
        Function = "Minutiae";
      }
      //If scribble button is clicked
      if (x > 1320 && x < 1400) {
        Function = "Scribble";
      }
      //If notes button is clicked
      if (x > 1430 && x < 1510) {
        Function = "Notes";
      }
    }
    //If opinion rect is selected
    if (Function == "Decision") {
      if (AnswerGiven == false) {
        if (y < height/2 + 130 && y > height/2 + 70) {
          if (x < width/2 - 220 && x > width/2 - 280) {
            Opinion = 1;
          }
          if (x < width/2 - 120 && x > width/2 - 180) {
            Opinion = 2;
          }
          if (x < width/2 - 20 && x > width/2 - 80) {
            Opinion = 3;
          }
          if (x > width/2 + 20 && x < width/2 + 80) {
            Opinion = 4;
          }
          if (x > width/2 + 120 && x < width/2 + 180) {
            Opinion = 5;
          }
          if (x > width/2 + 220 && x < width/2 + 280) {
            Opinion = 6;
          }
        }
      }

      //If continue button on opinion popup is clicked
      if (y < 790 && y > 710) {
        if (x < width/2 + 125 && x > width/2 - 125) {
          //If you gave an answer, the button will change functionality
          if (AnswerGiven) {
            if (Correct == false) {
              AnswerGiven = false;
              Opinion = 0;
            }
            if (Correct || Phase.equals("Comparison3")) {
              if (Phase.equals("Comparison1")) {
                row = Stats.getRow(7);
                Phase = "PatternScreen2";
              }
              if (Phase.equals("Comparison2")) {
                row = Stats.getRow(13);
                Phase = "StartFinal";
              }
              if (Phase.equals("Comparison3")) {
                row = Stats.getRow(19);
                Phase = "EndScreen";
              }
              row.setString(1, Notes);
              row = Stats.getRow(0);
              row.setString(1, Phase);
              //reset for next comparison
              for (int i = 0; i < scribble.size(); i=0) {
                scribble.remove(i);
              }
              Notes = "";
              AnswerGiven = false;
              Correct = false;
              Opinion = 0;
              Guesses = 1;
              Function = "Minutiae";
            }
          }
          //If the button is clicked to save the answer
          if (Opinion > 0 && AnswerGiven == false) {
            AnswerGiven = true;
            if (Phase.equals("Comparison1")) {
              row = Stats.getRow(6);
            }
            if (Phase.equals("Comparison2")) {
              row = Stats.getRow(12);
            }
            if (Phase.equals("Comparison3")) {
              row = Stats.getRow(18);
            }
            row.setInt(Guesses, Opinion);
            //saveTable(Stats, "data/Stats.csv");
            if (decisionCorrect()) {
              Correct = true;
            } else {
              Guesses++;
              Correct = false;
            }
          }
        }
      }
    }
    saveTable(Stats, "data/Stats.csv");
  }
  void KeyPressed(char k) {
    //Allow typing in the notes popup
    if (k == BACKSPACE && Notes.length() > 0) {
      Notes = Notes.substring(0, Notes.length()-1);
    } else {
      Notes += k;
    }
  }

  boolean onLatentPrint(float x, float y) {
    if (Function == "Minutiae") {
      if ((x >= xpos1 - size/2 && x <= xpos1 + size/2)) {
        if (y >= ypos - size/2 && y <= ypos + size/2) {
          return true;
        } else {
          return false;
        }
      } else {
        return false;
      }
    } else {
      return false;
    }
  }

  boolean onRolledPrint(float x, float y) {
    if (Function == "Minutiae") {
      if ((x >= xpos2 - size/2 && x <= xpos2 + size/2)) {
        if (y >= ypos - size/2 && y <= ypos + size/2) {
          return true;
        } else {
          return false;
        }
      } else {
        return false;
      }
    } else {
      return false;
    }
  }

  boolean decisionCorrect() {
    //Get the right answers of match/nomatch
    if (Phase.equals("Comparison1")) {
      row = Stats.getRow(4);
    }
    if (Phase.equals("Comparison2")) {
      row = Stats.getRow(10);
    }
    if (Phase.equals("Comparison3")) {
      row = Stats.getRow(16);
    }

    String Answer = row.getString(1);
    if ((Opinion >= 1 && Opinion <= 3 && Answer.equals("NoMatch")) || (Opinion >= 4 && Opinion
<= 6 && Answer.equals("Match"))) {
      return true;
    } else {
      return false;
    }
  }

  void Scribble(float x, float y) {
    if (mouseButton == LEFT && y < 900) {
      scribble.add(new Scribble(x, y, false));
    }
    //allow people to erase scribbles
    if (mouseButton == RIGHT) {
      strokeWeight(3);
      stroke(0);
      noFill();
      circle(x, y, 10);
      for (int i = 0; i<scribble.size(); i++) {
        Scribble singleScribble1 = scribble.get(i);
        if (x > singleScribble1.posx -8 && x < singleScribble1.posx +8 && y >
singleScribble1.posy - 8 && y < singleScribble1.posy + 8) {
          scribble.remove(i);
          if (i > 0) {
            Scribble singleScribble2 = scribble.get(i-1);
            singleScribble2.endpoint = true;
          }
        }
      }
    }
  }

  void ScribbleReleased(float x, float y) {
    //used to make scribble end
    if (Function == "Scribble") {
      scribble.add(new Scribble(x, y, true));
    }
  }

  void Popup() {
    if (Function == "Notes") {
      rectMode(CENTER);
      stroke(0, 200);
      fill(255, 200);
      rect(width/2, height/2, 1200, 750);
      fill(0);
      textSize(20);
      text(Notes, width/2, height/2, 1180, 720);
    }
  }
  void ScreenShot() {
    if (Function == "Screenshot1") {
      saveFrame("Comparison1");
      Function = "Decision";
    }
    if (Function == "Screenshot2") {
      saveFrame("Comparison2");
      Function = "Decision";
    }
    if (Function == "Screenshot3") {
      saveFrame("Comparison3");
      Function = "Decision";
    }
  }

  void Decision() {
    //Final decision popup
    if (Function == "Decision") {
      //Background
      fill(255, 100);
      rectMode(CENTER);
      //Main screen
      stroke(0);
      //Change color according to given answer
      if (Phase.equals("Comparison3")) {
        fill(200, 200);
      } else if (AnswerGiven == false) {
        fill(255, 200);
      } else if (Correct == true) {
        fill(0, 255, 0, 200);
      } else {
        fill(255, 0, 0, 200);
      }
      rect(width/2, height/2, 1200, 750, 20);
      fill(0);
      textAlign(CENTER, CENTER);
      textSize(60);
      //Display the correct texts
      if (AnswerGiven == false) {
        text("Wat is jouw uiteindelijke mening?", width/2, height/2-100);
      } else if (Phase.equals("Comparison3")) {
        text("Jouw antwoord is opgeslagen", width/2, height/2-120);
        text("ga verder", width/2, height/2-70);
      } else if (Correct == true) {
        text("Dat is juist! Goed gedaan", width/2, height/2-100);
      } else {
        text("Helaas, dat is niet juist", width/2, height/2-100);
      }

      textSize(30);
      text("Zeer waarschijnlijk", 480, height/2 +75);
      text("andere bronnen", 480, height/2 +115);
      text("Zeer waarschijnlijk", 1330, height/2 +75);
      text("dezelfe bron", 1330, height/2 +115);

      //continue button
      textAlign(CENTER);
      textSize(40);
      stroke(0);
      rectMode(CENTER);
      fill(0, 255, 0);
      if (AnswerGiven == false) {
        //if (Opinion == 0) {
        fill(80);
        //}
      } else if (Correct == false) {
        fill(255, 0, 0);
      }
      if (Phase.equals("Comparison3")) {
        fill(200);
      }
      rect(width/2, 750, 250, 80, 20);
      fill(255);

      if (Phase.equals("Comparison3") || Correct || AnswerGiven == false) {
        text("Continue", width/2, 760);
      } else if (Correct == false) {
        textSize(30);
        text("Probeer opnieuw", width/2, 760);
      }

      //Draw option boxes and only fill if its ticked
      if (Opinion == 1) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 - 250, height/2 + 100, 60, 60, 10);
      if (Opinion == 2) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 - 150, height/2 + 100, 60, 60, 10);
      if (Opinion == 3) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 - 50, height/2 + 100, 60, 60, 10);
      if (Opinion == 4) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 + 50, height/2 + 100, 60, 60, 10);
      if (Opinion == 5) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 + 150, height/2 + 100, 60, 60, 10);
      if (Opinion == 6) {
        fill(150);
      } else {
        noFill();
      }
      rect(width/2 + 250, height/2 + 100, 60, 60, 10);
    }
  }
}
```

```
class DeleteMinutiae {
  Table MarkedMinutiae;
  String Phase;
  float posx;
  float posy;
  float margin = 5;
  DeleteMinutiae() {
  }

  void Delete(String p, float x, float y) {
    Phase = p;
    //Load correct table
    if (Phase.equals("LatentPrint1") || (Phase.equals("Comparison1") &&
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae1.csv");
    }
    if (Phase.equals("RolledPrint1")|| (Phase.equals("Comparison1") &&
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae2.csv");
    }
    if (Phase.equals("LatentPrint2") || (Phase.equals("Comparison2") &&
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae3.csv");
    }
    if (Phase.equals("RolledPrint2")|| (Phase.equals("Comparison2") &&
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae4.csv");
    }
    if (Phase.equals("LatentPrint3") || (Phase.equals("Comparison3") &&
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae5.csv");
    }
    if (Phase.equals("RolledPrint3")|| (Phase.equals("Comparison3") &&
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae6.csv");
    }
    if (MarkedMinutiae.getRowCount() > 1) {
      for (int i = 1; i < MarkedMinutiae.getRowCount(); i++) {
        TableRow row = MarkedMinutiae.getRow(i);
        //Use the right xpos corrector
        if ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) && comparisonScreen.onLatentPrint(x, y)) {
          posx = row.getFloat(0) + comparisonScreen.xpos1;
          posy = row.getFloat(1) + comparisonScreen.ypos;
        }
        if ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) && comparisonScreen.onRolledPrint(x, y)) {
          posx = row.getFloat(0) + comparisonScreen.xpos2;
          posy = row.getFloat(1) + comparisonScreen.ypos;
        }
        if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
          posx = row.getFloat(0) + analysisScreen.xpos;
          posy = row.getFloat(1) + analysisScreen.ypos;
        }
        //If minutiae is rightclicked, set the minutiae to deleted
        if (x > posx - margin && x < posx + margin) {
          if (y > posy - margin && y < posy + margin) {
            row.setString(6, "Yes");
            //Save in the right table
            if (Phase.equals("LatentPrint1") || (Phase.equals("Comparison1") &&
comparisonScreen.onLatentPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae1.csv");
            }
            if (Phase.equals("RolledPrint1")|| (Phase.equals("Comparison1") &&
comparisonScreen.onRolledPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae2.csv");
            }
            if (Phase.equals("LatentPrint2") || (Phase.equals("Comparison2") &&
comparisonScreen.onLatentPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae3.csv");
            }
            if (Phase.equals("RolledPrint2")|| (Phase.equals("Comparison2") &&
comparisonScreen.onRolledPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae4.csv");
            }
            if (Phase.equals("LatentPrint3") || (Phase.equals("Comparison3") &&
comparisonScreen.onLatentPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae5.csv");
            }
            if (Phase.equals("RolledPrint3")|| (Phase.equals("Comparison3") &&
comparisonScreen.onRolledPrint(x, y))) {
              saveTable(MarkedMinutiae, "data/MarkedMinutiae6.csv");
            }
          }
        }
      }
    }
  }
}
```

```
class DirectFeedback {
  float xpos;
  float ypos;
  boolean answer;
  boolean position;
  int life = 255;
  DirectFeedback(float x, float y, boolean ans, boolean pos) {
    xpos = x;
    ypos = y;
    answer = ans;
    position = pos;
  }

  void Display() {
    noStroke();
    //Correct is green, semi-correct is yellow, incorrect is red
    if (position) {
      fill(255, 255, 0, life/2);
      if (answer) {
        fill(0, 255, 0, life/2);
      }
    } else {
      fill(255, 0, 0, life/2);
    }
    circle(xpos, ypos, life/7);
    life -= 2;
  }

  boolean isDead() {
    if (life<=0) {
      return true;
    } else {
      return false;
    }
  }
}
```

```
class DrawMinutiae {
  Table MarkedMinutiae;
  String Phase;
  float x;
  float y;
  String label;
  String tolerance;
  float degrees;
  String deleted;

  color t = color(0, 255, 0);
  float size = 10;

  DrawMinutiae() {
  }

  void Run(String p) {
    Phase = p;

    //Load the right table
    if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
      if (Phase.equals("LatentPrint1")) {
        MarkedMinutiae = loadTable("MarkedMinutiae1.csv");
      }
      if (Phase.equals("RolledPrint1")) {
        MarkedMinutiae = loadTable("MarkedMinutiae2.csv");
      }
      if (Phase.equals("LatentPrint2")) {
        MarkedMinutiae = loadTable("MarkedMinutiae3.csv");
      }
      if (Phase.equals("RolledPrint2")) {
        MarkedMinutiae = loadTable("MarkedMinutiae4.csv");
      }
      if (Phase.equals("LatentPrint3")) {
        MarkedMinutiae = loadTable("MarkedMinutiae5.csv");
      }
      if (Phase.equals("RolledPrint3")) {
        MarkedMinutiae = loadTable("MarkedMinutiae6.csv");
      }
      //Get all the info of the minutiae from the table
      if (MarkedMinutiae.getRowCount() > 1) {
        for (int i = 1; i < MarkedMinutiae.getRowCount(); i++) {
          TableRow row = MarkedMinutiae.getRow(i);
          deleted = row.getString(6);
          if (deleted.equals("No")) {
            x = row.getFloat(0) + analysisScreen.xpos;
            y = row.getFloat(1) + analysisScreen.ypos;
            label = row.getString(2);
            tolerance = row.getString(3);
            degrees = row.getFloat(4);
            Display(x, y, label, tolerance, degrees);
          }
        }
      }
    }

    //If in comparison phase, draw minutiae for both prints
    if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
      if (Phase.equals("Comparison1")) {
        MarkedMinutiae = loadTable("MarkedMinutiae1.csv");
      }
      if (Phase.equals("Comparison2")) {
        MarkedMinutiae = loadTable("MarkedMinutiae3.csv");
      }
      if (Phase.equals("Comparison3")) {
        MarkedMinutiae = loadTable("MarkedMinutiae5.csv");
      }
      if (MarkedMinutiae.getRowCount() > 1) {
        for (int i = 1; i < MarkedMinutiae.getRowCount(); i++) {
          TableRow row = MarkedMinutiae.getRow(i);
          deleted = row.getString(6);
          if (deleted.equals("No")) {
            x = row.getFloat(0) + comparisonScreen.xpos1;
            y = row.getFloat(1) + comparisonScreen.ypos;
            label = row.getString(2);
            tolerance = row.getString(3);
            degrees = row.getFloat(4);
            Display(x, y, label, tolerance, degrees);
          }
        }
      }
      if (Phase.equals("Comparison1")) {
        MarkedMinutiae = loadTable("MarkedMinutiae2.csv");
      }
      if (Phase.equals("Comparison2")) {
        MarkedMinutiae = loadTable("MarkedMinutiae4.csv");
      }
      if (Phase.equals("Comparison3")) {
        MarkedMinutiae = loadTable("MarkedMinutiae6.csv");
      }
      if (MarkedMinutiae.getRowCount() > 1) {
        for (int i = 1; i < MarkedMinutiae.getRowCount(); i++) {
          TableRow row = MarkedMinutiae.getRow(i);
          deleted = row.getString(6);
          if (deleted.equals("No")) {
            x = row.getFloat(0) + comparisonScreen.xpos2;
            y = row.getFloat(1) + comparisonScreen.ypos;
            label = row.getString(2);
            tolerance = row.getString(3);
            degrees = row.getFloat(4);
            Display(x, y, label, tolerance, degrees);
          }
        }
      }
    }
  }

  void Display(float x, float y, String label, String tolerance, float degrees) {
    if (tolerance.equals("Green")) {
      t = color(0, 255, 0);
    } else if (tolerance.equals("Yellow")) {
      t = color(255, 255, 0);
    } else if (tolerance.equals("Red")) {
      t = color(255, 0, 0);
    } else if (tolerance.equals("Orange")) {
      t = color(255, 150, 0);
    }

    pushMatrix();
    translate(x, y);
    fill(t);
    strokeWeight(size/3);
    stroke(t);
    rotate(radians(degrees));
    rectMode(CORNER);
    if (label.equals("Bifurcation")) {
      circle(0, 0, size);
    }
    if (label.equals("Ending")) {
      rect(-(size/2), -(size/2), size, size);
    }
    if (label.equals("Island")) {
      triangle(-(size/2), -(size/2), (size/2), -(size/2), 0, (size/2));
    }
    if (label.equals("Unclear")) {
      line(-3, 0, 3, 0);
    }
    if (label.equals("Island") == false) {
      line(0, 0, 0, -20);
    }
    popMatrix();
  }

  void Mouse(float x, float y, String label, String tolerance, float degrees) {
    //display a minutiae sign at mouseX and mouseY
    if (tolerance.equals("Green")) {
      t = color(0, 255, 0);
    } else if (tolerance.equals("Yellow")) {
      t = color(255, 255, 0);
    } else if (tolerance.equals("Red")) {
      t = color(255, 0, 0);
    } else if (tolerance.equals("Orange")) {
      t = color(255, 150, 0);
    }

    pushMatrix();
    translate(x, y);
    fill(t);
    strokeWeight(size/3);
    stroke(t);
    rotate(radians(degrees));
    rectMode(CORNER);
    if (label.equals("Bifurcation")) {
      circle(0, 0, size);
    }
    if (label.equals("Ending")) {
      rect(-(size/2), -(size/2), size, size);
    }
    if (label.equals("Island")) {
      triangle(-(size/2), -(size/2), (size/2), -(size/2), 0, (size/2));
    }
    if (label.equals("Unclear")) {
      line(-3, 0, 3, 0);
    }
    if (label.equals("Island") == false) {
      line(0, 0, 0, -20);
    }
    popMatrix();
  }
}
```

```
class FinalScreens {
  Table Stats;
  String Phase;

  FinalScreens() {
  }

  void StartFinal() {
    textAlign(LEFT);
    textSize(40);
    stroke(0);
    rectMode(CENTER);
    fill(0, 255, 0);
    rect(1665, 950, 250, 80, 20);
    fill(255);
    text("Continue", 1580, 960);

    fill(0);
    textAlign(CENTER);
    text("Je hebt de twee oefenrondes afgerond", width/2, 300);
    text("Je gaat nu jouw skills gebruiken voor een laatste set,", width/2, 520);
    text("maar deze keer zonder feedback.", width/2, 600);
    textSize(100);
    text("Goed gedaan!", width/2, 200);
    textSize(80);
    text("Succes!", width/2, 800);
  }

  void End() {
    textAlign(LEFT);
    textSize(40);
    stroke(0);
    rectMode(CENTER);
    fill(0, 255, 0);
    rect(1665, 950, 250, 80, 20);
    fill(255);
    text("Close", 1600, 960);

    fill(0);
    textAlign(CENTER);
    text("Sla alstjeblieft alle documenten op als een zip-file en stuur ze naar:", width/2,
620);
    text("d.elskamp@student.utwente.nl", width/2, 700);
    textSize(100);
    text("Goed gedaan!", width/2, 300);
    textSize(80);
    text("Heel erg bedankt voor het meewerken!", width/2, 400);
  }

  void MouseClicked(String p, float x, float y) {
    Phase = p;
    //If continue button is clicked, move to next phase, or close program
    if (y > 910 && y < 990) {
      if (x < 1790 && x > 1540) {
        if (Phase.equals("StartFinal")) {
          Phase = "PatternScreen3";
          Stats = loadTable("Stats.csv");
          TableRow row = Stats.getRow(0);
          row.setString(1, Phase);
          saveTable(Stats, "data/Stats.csv");
        }
        if (Phase.equals("EndScreen")) {
          exit();
        }
      }
    }
  }
}
```

```
class MarkMinutiae {
  CheckMinutiae checkMinutiae;
  DrawMinutiae drawMinutiae;

  String Phase;
  float   degrees;
  float   rotation;
  String  label     = "Bifurcation";
  String  tolerance = "Green";

  MarkMinutiae() {
    checkMinutiae = new CheckMinutiae();
    drawMinutiae = new DrawMinutiae();
  }

  void Select(char q) {
    // Selecting tolerance (G, Y & R)
    if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
      if (q == &apos;G&apos; || q == &apos;g&apos;) {
        tolerance = "Green";
      }
      if (q == &apos;Y&apos; || q == &apos;y&apos;) {
        tolerance = "Yellow";
      }
      if (q == &apos;R&apos; || q == &apos;r&apos;) {
        tolerance = "Red";
      }
    } else{
     tolerance = "Orange";
    }

    // Selecting label (1, 2, 3 & 4)
    if (q == &apos;1&apos;) {
      label = "Bifurcation";
    }
    if (q == &apos;2&apos;) {
      label = "Ending";
    }
    if (q == &apos;3&apos;) {
      label = "Island";
    }
    if (q == &apos;4&apos;) {
      label = "Unclear";
    }

    // Adjusting orientation (RIGHT & LEFT)
    if (q == &apos;,&apos;) {
      if (rotation < 3) {
        rotation += 3;
      } else {
        rotation++;
      }
      degrees -= rotation;
    }
    if (q == &apos;.&apos;) {
      if (rotation < 3) {
        rotation += 3;
      } else {
        rotation++;
      }
      rotation++;
      degrees += rotation;
    }
  }

  void KeyReleased() {
    rotation = 0;
  }

  void MouseClickedLeft(String p, float x, float y) {
    Phase = p;
    if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
      tolerance = "Orange";
    }
    checkMinutiae.Check(Phase, x, y, label, tolerance, degrees%360);
  }

  void Mouse(String p, float x, float y) {
    Phase = p;
    if (Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) {
      tolerance = "Orange";
    }
    drawMinutiae.Mouse(x, y, label, tolerance, degrees%360);
  }
}
```

```
class PatternScreen {
```

```
Table Stats;
TableRow row;
PImage Latent1 = loadImage("LatentPrint1.png");
PImage Rolled1 = loadImage("RolledPrint1.png");
PImage Latent2 = loadImage("LatentPrint2.png");
PImage Rolled2 = loadImage("RolledPrint2.png");
PImage Latent3 = loadImage("LatentPrint3.png");
PImage Rolled3 = loadImage("RolledPrint3.png");
String username;
String Phase;
String Latent = "";
String Rolled = "";
String Pattern;
boolean Continued = false;
boolean Correct;
int Guesses = 1;

PatternScreen() {
}

void Run(String p) {
  Phase = p;
  //Get username to display
  Stats = loadTable("Stats.csv");
  row = Stats.getRow(1);
  username = row.getString(1);
  textAlign(LEFT, TOP);
  textSize(40);
  fill(255);
  text(username, 10, 10);

  imageMode(CORNER);
  //Display right images
  if (Phase.equals("PatternScreen1")) {
    image(Latent1, 200, 120, 600, 600);
    image(Rolled1, 1000, 120, 600, 600);
  }
  if (Phase.equals("PatternScreen2")) {
    image(Latent2, 200, 120, 600, 600);
    image(Rolled2, 1000, 120, 600, 600);
  }
  if (Phase.equals("PatternScreen3")) {
    image(Latent3, 200, 120, 600, 600);
    image(Rolled3, 1000, 120, 600, 600);
  }

  rectMode(CORNER);
  strokeWeight(3);
  textAlign(CENTER, CENTER);
  textSize(40);
  //Latent choices
  if (Latent == "Arch") {
    fill(255);
  } else {
    fill(150);
  }
  rect(200, 740, 196, 100, 10);
  if (Latent == "Loop") {
    fill(255);
  } else {
    fill(150);
  }
  rect(402, 740, 196, 100, 10);
  if (Latent == "Whorl") {
    fill(255);
  } else {
    fill(150);
  }
  rect(604, 740, 196, 100, 10);
  fill(0);
  text("Arch", 298, 790);
  text("Loop", 500, 790);
  text("Whorl", 702, 790);

  //Rolled choices
  if (Rolled == "Arch") {
    fill(255);
  } else {
    fill(150);
  }
  rect(1000, 740, 196, 100, 10);
  if (Rolled == "Loop") {
    fill(255);
  } else {
    fill(150);
  }
  rect(1202, 740, 196, 100, 10);
  if (Rolled == "Whorl") {
    fill(255);
  } else {
    fill(150);
  }
  rect(1404, 740, 196, 100, 10);
  fill(0);
  text("Arch", 1098, 790);
  text("Loop", 1300, 790);
  text("Whorl", 1502, 790);

  //Continue
  textAlign(LEFT);
  textSize(40);
  stroke(0);
  rectMode(CENTER);
  if (Latent != "" && Rolled != "") {
    fill(0, 255, 0);
  } else {
    fill(80);
  }
  rect(1665, 950, 250, 80, 20);
  fill(255);
  text("Continue", 1580, 960);

  if (Continued) {
    PopUp();
  }
}

void MouseClicked(String p, float x, float y) {
  Phase = p;
  if (Continued == false) {
    //Selecting pattern options
    if (y >= 770 && y <= 870) {
      if (x >= 200 && x<= 396) {
        Latent = "Arch";
      }
      if (x >= 402 && x<= 598) {
        Latent = "Loop";
      }
      if (x >= 604 && x<= 800) {
        Latent = "Whorl";
      }
      if (x >= 1000 && x<= 1196) {
        Rolled = "Arch";
      }
      if (x >= 1202 && x<= 1398) {
        Rolled = "Loop";
      }
      if (x >= 1404 && x<= 1600) {
        Rolled = "Whorl";
      }
    }
    //Selecting continue
    if (Latent != "" && Rolled != "") {
      if (x < 1790 && x > 1540 && y > 910 && y < 990) {
        //Get the Pattern answer
        if (Phase.equals("PatternScreen1")) {
          row = Stats.getRow(3);
          Pattern = row.getString(1);
        }
        if (Phase.equals("PatternScreen2")) {
          row = Stats.getRow(9);
          Pattern = row.getString(1);
```

```
     }
     Continued = true;
     //Save guesses in table
     if (Phase.equals("PatternScreen1")) {
      row = Stats.getRow(5);
     }
     if (Phase.equals("PatternScreen2")) {
      row = Stats.getRow(11);
     }
     if (Phase.equals("PatternScreen3")) {
      row = Stats.getRow(17);
     }
     row.setString(Guesses, (Latent + ", " + Rolled));
     saveTable(Stats, "data/Stats.csv");
     if (Phase.equals("PatternScreen1") || Phase.equals("PatternScreen2")) {
      if (Latent.equals(Pattern) && Rolled.equals(Pattern)) {
       Correct = true;
       Guesses = 1;
      } else {
       Correct = false;
       Guesses++;
      }
     }
    }
   }
  }
  //If in popup screen
  else {
   //If continue button clicked
   if (x < width/2 + 400 && x > width/2 - 400 && y > height/2 + 80  && y < height/2 + 160)
{
    row = Stats.getRow(0);
    if (Correct || Phase.equals("PatternScreen3")) {
     if (Phase.equals("PatternScreen1")) {
      row.setString(1, "LatentPrint1");
     }
     if (Phase.equals("PatternScreen2")) {
      row.setString(1, "LatentPrint2");
     }
     if (Phase.equals("PatternScreen3")) {
      row.setString(1, "LatentPrint3");
     }
     saveTable(Stats, "data/Stats.csv");
     markMinutiae.tolerance = "Green";
     //Reset for new PatternScreen
     Latent = "";
     Rolled = "";
     Continued = false;
    } else {
     Continued = false;
    }
   }
  }
 }

 void PopUp() {
  //main popup
  if (Phase.equals("PatternScreen3")) {
   fill(200, 200);
  } else if (Correct) {
   fill(0, 255, 0, 200);
  } else {
   fill(255, 0, 0, 200);
  }
  rect(width/2, height/2, 800, 400, 10);
  //continue button
  if (Phase.equals("PatternScreen3")) {
   fill(200);
  } else if (Correct) {
   fill(0, 255, 0);
  } else {
   fill(255, 0, 0);
  }
  rect(width/2, height/2 + 120, 250, 80, 20);

  fill(0);
```

```
    textAlign(CENTER);
    if (Phase.equals("PatternScreen3")) {
      textSize(50);
      text("Jouw antwoord is opgeslagen, ", width/2, height/2 - 100);
      textSize(40);
      text("Ga verder naar de volgende pagina", width/2, height/2);
      text("Continue", width/2, height/2 + 130);
    } else if (Correct) {
      textSize(60);
      text("Goed gedaan, ", width/2, height/2 - 100);
      textSize(40);
      text("Je hebt de juiste patronen herkend", width/2, height/2);
      text("Continue", width/2, height/2 + 130);
    } else {
      textSize(60);
      text("Helaas, ", width/2, height/2 - 100);
      textSize(40);
      text("Je hebt de juiste patronen niet herkend", width/2, height/2);
      textSize(30);
      text("Probeer opnieuw", width/2, height/2 + 130);
    }
  }
}
```

```
class SaveMinutiae {
  Table MarkedMinutiae;
  String Phase;
  String Answer;
  float degrees;

  SaveMinutiae() {
  }

  void Save(String p, float x, float y, String l, String t, float d, boolean answer, boolean 
pos) {
    Phase = p;
    if (pos) {
      Answer = "SemiCorrect";
      if (answer) {
        Answer = "Correct";
      }
    } else {
      Answer = "Incorrect";
    }
    if (d >= 0) {
      degrees = d;
    } else {
      degrees = d + 360;
    }
    //Load right table
    if (Phase.equals("LatentPrint1") || (Phase.equals("Comparison1") && 
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae1.csv");
    }
    if (Phase.equals("RolledPrint1")|| (Phase.equals("Comparison1") && 
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae2.csv");
    }
    if (Phase.equals("LatentPrint2") || (Phase.equals("Comparison2") && 
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae3.csv");
    }
    if (Phase.equals("RolledPrint2")|| (Phase.equals("Comparison2") && 
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae4.csv");
    }
    if (Phase.equals("LatentPrint3") || (Phase.equals("Comparison3") && 
comparisonScreen.onLatentPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae5.csv");
    }
    if (Phase.equals("RolledPrint3")|| (Phase.equals("Comparison3") && 
comparisonScreen.onRolledPrint(x, y))) {
      MarkedMinutiae = loadTable("MarkedMinutiae6.csv");
    }
    int i = MarkedMinutiae.getRowCount();
    TableRow row = MarkedMinutiae.getRow(i);
    //Use the right xpos corrector
    if ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) && comparisonScreen.onLatentPrint(x, y)) {
      row.setFloat(0, x - comparisonScreen.xpos1);
      row.setFloat(1, y - comparisonScreen.ypos);
    }
    if ((Phase.equals("Comparison1") || Phase.equals("Comparison2") ||
Phase.equals("Comparison3")) && comparisonScreen.onRolledPrint(x, y)) {
      row.setFloat(0, x - comparisonScreen.xpos2);
      row.setFloat(1, y - comparisonScreen.ypos);
    }
    if (Phase.equals("LatentPrint1") || Phase.equals("RolledPrint1") ||
Phase.equals("LatentPrint2") || Phase.equals("RolledPrint2") || Phase.equals("LatentPrint3")
|| Phase.equals("RolledPrint3")) {
      row.setFloat(0, x - analysisScreen.xpos);
      row.setFloat(1, y - analysisScreen.ypos);
    }
    row.setString(2, l);
    row.setString(3, t);
    row.setFloat(4, degrees);
    row.setString(5, Answer);
    row.setString(6, "No");
    //Save in the right table
    if (Phase.equals("LatentPrint1") || (Phase.equals("Comparison1") &&
comparisonScreen.onLatentPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae1.csv");
    }
    if (Phase.equals("RolledPrint1") || (Phase.equals("Comparison1") &&
comparisonScreen.onRolledPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae2.csv");
    }
    if (Phase.equals("LatentPrint2") || (Phase.equals("Comparison2") &&
comparisonScreen.onLatentPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae3.csv");
    }
    if (Phase.equals("RolledPrint2") || (Phase.equals("Comparison2") &&
comparisonScreen.onRolledPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae4.csv");
    }
    if (Phase.equals("LatentPrint3") || (Phase.equals("Comparison3") &&
comparisonScreen.onLatentPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae5.csv");
    }
    if (Phase.equals("RolledPrint3") || (Phase.equals("Comparison3") &&
comparisonScreen.onRolledPrint(x, y))) {
      saveTable(MarkedMinutiae, "data/MarkedMinutiae6.csv");
    }
  }
}
```

```
class Scribble {
  float posx;
  float posy;
  boolean endpoint;

  Scribble(float x, float y, boolean e) {
    posx = x;
    posy = y;
    endpoint = e;
  }
}
```

```
class VideoScreen {
  Movie movie;
  Table Stats;
  float ScrollBall;
  boolean paused = false;
  boolean endReached = false;
  String username;
  Boolean Continued = false;

  VideoScreen() {
    movie = new Movie(app, "Vid.mp4");
  }
```

```
void Run() {
  //Load username to display
  Stats = loadTable("Stats.csv");
  TableRow row = Stats.getRow(1);
  username = row.getString(1);
  textAlign(LEFT, TOP);
  textSize(40);
  fill(255);
  text(username, 10, 10);

  if (paused) {
    movie.pause();
  } else {
    movie.play();
  }
  imageMode(CENTER);
  image(movie, width/2, height/2, 1280, 720);

  //Movie slider with time in movie
  ScrollBall = movie.time()*(1260/movie.duration()) + width/2 - 630;
  if (movie.time() == movie.duration()) {
    endReached = true;
  }
  //Show video line and slider
  stroke(255);
  strokeWeight(3);
  line(width/2 - 630, 820, width/2 + 630, 820);
  stroke(255, 0, 0);
  line(width/2 - 630, 820, ScrollBall, 820);
  fill(255, 0, 0);
  circle(ScrollBall, 820, 10);

  fill(255);
  rectMode(CENTER);
  noStroke();
  //pause play sign
  if (paused) {
    triangle(width/2 + 10, 910, width/2 - 10, 890, width/2 - 10, 930);
  } else {
    rect(width/2 - 10, 910, 10, 40);
    rect(width/2 + 10, 910, 10, 40);
  }
  //fast forward
  triangle( width/2 + 70, 910, width/2 + 50, 890, width/2 + 50, 930);
  triangle( width/2 + 90, 910, width/2 + 70, 890, width/2 + 70, 930);
  //fast backward
  triangle( width/2 - 70, 910, width/2 - 50, 890, width/2 - 50, 930);
  triangle( width/2 - 90, 910, width/2 - 70, 890, width/2 - 70, 930);

  textAlign(LEFT);
  textSize(40);
  stroke(0);

  //continue button
  if (endReached) {
    fill(0, 255, 0);
  } else {
    fill(80);
  }
  rect(1665, 950, 250, 80, 20);
  fill(255);
  text("Continue", 1580, 960);

  //Warning and cancel sign
  if (Continued) {
    fill(200);
    rect(1600, 800, 380, 200, 20);
    fill(0);
    textSize(20);
    text("Weet je zeker dat je wilt doorgaan?", 1420, 750);
    text("Je kan niet terug naar deze pagina", 1420, 785);
    fill(255, 0, 0);
    rect(1665, 860, 250, 80, 20);
    fill(255);
    textSize(40);
```

```
      text("Cancel", 1600, 870);
    }
  }

  void MouseClicked(float x, float y) {
    //click on bar to jump to time
    if (x >= width/2 - 630 && x <= width/2 + 630 && y >= 817 && y <= 823) {
      float ratio = (mouseX-(width/2 - 630)) / 1260.0;
      movie.jump(ratio * movie.duration());
    }
    //click on video buttons
    if (y >= 890 && y <= 930) {
      //pause and play
      if (x > width/2 - 15 && x < width/2 + 15) {
        if (paused) {
          paused = false;
        } else {
          paused = true;
        }
      }
      //backward
      if (x > width/2 - 90 && x < width/2 - 50) {
        movie.jump(0);
      }
      //forward
      if (x > width/2 + 50 && x < width/2 + 90) {
        movie.jump(movie.duration());
      }
    }
    //continue (only possible if people reached the end of the video)
    if (x < 1790 && x > 1540 && y > 910 && y < 990) {
      if (endReached) {
        if (Continued) {
          Continued = false;
          TableRow row = Stats.getRow(0);
          row.setString(1, "PatternScreen1");
          saveTable(Stats, "data/Stats.csv");
        } else {
          Continued = true;
        }
      }
    }
    if (x < 1790 && x > 1540 && y > 820 && y < 900) {
      Continued = false;
    }
  }
}

void movieEvent(Movie m) {
  m.read();
}
```

```
class WelcomeScreen {
  Table Stats;
  boolean typing = false;
  String username = "";
  int timer;

  WelcomeScreen() {
    Stats = loadTable("Stats.csv");
  }

  void Run() {
    textAlign(CENTER, CENTER);
    fill(255);
    textSize(150);
    text("- Welkom -", width/2, 200);
    textSize(50);
    text("bij deze vingerafdruk vergelijkingstraining!", width/2, 350);
    textSize(30);
    text("Voer hier je gebruikersnaam in", width/2, 540);
    textSize(25);
    text("(Dit hoeft niet je eigen naam te zijn)", width/2, 580);

    rectMode(CENTER);
    rect(width/2, 650, 500, 80, 10);
    fill(0);
    textAlign(LEFT);
    textSize(40);
    text(username, 660, 665);

    //Typing line
    strokeWeight(3);
    if (typing && timer < 50) {
      line(660 + textWidth(username), 670, 660 + textWidth(username), 630);
    }
    timer++;
    if (timer == 100) {
      timer = 0;
    }
    //If people typed a username, the continue button chamges color
    if (username == "") {
      fill(80);
    } else {
      fill(0, 255, 0);
    }
    rect(1665, 950, 250, 80, 20);
    fill(0);
    text("Continue", 1580, 960);
  }

  void MouseClicked(float x, float y) {
    if (x < width/2 + 250 && x > width/2 - 250) {
      if (y <  690 && y > 610) {
        typing = true;
      } else {
        typing = false;
      }
    } else {
      typing = false;
    }
    if (x < 1790 && x > 1540 && y > 910 && y < 990) {
      if (username != "") {
        TableRow row = Stats.getRow(0);
        row.setString(1, "VideoScreen");
        row = Stats.getRow(1);
        row.setString(1, username);
        saveTable(Stats, "data/Stats.csv");
      }
    }
  }

  void Typing(char k) {
    if (typing && textWidth(username) < 460) {
      username += k;
    }
  }
  void BackSpace() {
    if (username.length() > 0) {
      username = username.substring(0, username.length()-1);
    }
  }
}
```

Appendix C

The Dutch script of the explanatory video implemented in the application.

Scene 1:

welkom bij deze vingerafdruk vergelijkingstraining. Tijdens deze training ga jij leren hoe je vingerafdrukken kan analyseren en hoe je kan ontdekken of twee vingerafdrukken van dezelfde vinger afkomstig zijn. Vergeet niet het toestemmingsformulier te lezen en te ondertekenen voordat je verder gaat met de training.

Verder is het erg belangrijk dat je de training individueel volgt, dus zonder hulp van anderen of het internet. Aangezien dit zeer waarschijnlijk de eerste keer is dat je je gaat verdiepen in vingerafdrukken is het logisch dat je niet alles in een keer goed kan doen. Voor mij is het dan ook interessanter om te zien waar de knelpunten liggen, in plaats van dat je gaat samenwerken met anderen of extra informatie opzoekt op het internet. Mocht je toch vragen hebben over het programma, of over de training, dan kun je me altijd een e-mail of een berichtje sturen.

Scene 2:

Laten we nu snel beginnen met de training.

Wanneer je vingerafdrukken wilt gaan vergelijken kijken we altijd eerst naar het algemene patroon. Er zijn drie verschillende soorten patronen en wanneer de twee vingerafdrukken niet hetzelfde patroon hebben weet je al meteen dat deze vingerafdrukken niet identiek zijn. De drie patronen zijn: de loop, waar de lijnen de vingerafdruk binnen komen, richting het centrum gaan en via dezelfde kant de vingerafdruk weer verlaten. De Arch, waar de lijnen de vingerafdruk binnen komen, richting het centrum gaan en aan de andere kant de vingerafdruk verlaten. En als laatste, de Whorl, waar de lijnen in het centrum een cirkel of ovaal vormen. In het programma kan je simpelweg de patronen selecteren en op de continue knop drukken.

Wanneer we gezien hebben dat alle twee de vingerafdrukken hetzelfde patroon hebben gaan we de vingerafdrukken 1 voor 1 in meer detail bestuderen. Hierbij gaan we op zoek naar opvallende kenmerken die we Minutiae noemen. In dit programma kan je verschillende soorten minutiae markeren. Allereerst de bifurcation, oftewel de splitsing. Bij een bifurcation volg je de lijn en zie je dat deze opsplitst in 2 lijnen. Met de 1 toets op je toetsenbord selecteer je het bifurcation label en met je pijltjestoetsen draai je de staart van je Minutiae. De staart plaats je met de lijn mee om de richting van de minutiae aan te geven. Wanneer je denkt dat de minutiae goed staat kan je hem vastzetten met je linkermuisknop.

De tweede soort is de ridge ending, oftwel het einde van een lijn. Wanneer je een lijn volgt zie je dat hij plotseling ophoudt en zijn ruimte wordt vaak opgevuld door de omringende lijnen. Met je 2 toets op je toetsenbord selecteer je ridge ending en je draait de staart weer in de goede richting. Wanneer je denkt dat je goed staat zet je hem weer vast met je linker muisknop.

De derde soort is de Island, wat simpelweg een stip in de vingerafdruk weergeeft. Bij de island hoef je geen staart aan te geven, aangezien het geen richting volgt.

Als laatste hebben we nog de unknown. Deze markering kan je eventueel gebruiken wanneer er wel een opmerkelijk punt zit, maar je niet weet welk label er bij past.

Naast dat je verschillende labels kan selecteren, kan je ook verschillende kleuren gebruiken. Met de letters G, Y en R op je toetsenbord kan je kiezen tussen groene, gele en rode markeringen. Groen betekent dat je veel waarde hecht aan dit punt, omdat je duidelijk kan zien dat er een minutiae zit en welk label er bij hoort. Bij geel weet je het heel iets minder zeker en dit punt zal je dus ook iets minder zwaar meenemen in je eindconclusie. Een rode markering betekent dat je de minutiae wel herkent, maar dat dit punt maar licht mee zal wegen omdat je er niet heel erg zeker van bent.

Wanneer je denkt dat je een fout hebt gemaakt, of je wil een markering aanpassen, dan kan je een markering verwijderen met je rechter muisknop. Pas wanneer je denkt dat je alle minutiae hebt gemarkeerd ga je door naar de volgende pagina.

Scene 3:

Wanneer je de twee vingerafdrukken apart geanalyseerd hebt ga je je markeringen vergelijken. Mocht je nog extra Minutiae vinden in de vingerafdrukken, omdat je ze bijvoorbeeld bij de ene wel gevonden had, maar bij de andere niet, dan kan je deze nog steeds markeren. Dit gebeurt in een oranje kleur, omdat dit punt blijkbaar in een losse analyse niet meteen herkenbaar was.

In deze fase kan je ook tekenen om dingen duidelijker te maken. Om te tekenen druk je op deze knop, en gebruik je je linker muisknop. Wanneer je een tekening wil verwijderen kan je je rechter muisknop gebruiken als gum.

In de notities kan je eventueel een conclusie schrijven om jouw analyses te verduidelijken.

Om de vingerafdrukken te goed te vergelijken ga je alle minutiae langs om te kijken of ze overeen komen. De beste manier om dit te doen is om de lijnen tussen de minutiae te tellen en zo te bevestigen of ze op dezelfde plek zitten. In deze fase wegen je groene markeringen dus ook zwaarder mee dan bijvoorbeeld je rode markeringen, omdat je hier minder zeker van was, maar let wel: 1 duidelijk verschil kan al genoeg zijn om te weten dat de afdrukken niet identiek zijn.

Wanneer je hier klaar bent druk je op continue om je uiteindelijke mening te geven. Denk je dat de twee afdrukken van dezelfde vinger afkomstig zijn of niet? En hoe zeker ben je hier van? Wanneer je dit hebt ingevuld ben je klaar met je eerste vergelijking en kan je door naar de volgende.

Scene 4.1 (Task Feedback):

Als laatste wil ik je nog een snelle uitleg geven over de feedback die jij gaat ontvangen tijdens de eerste twee vergelijkingen. Jij bent namelijk ingedeeld in de zogenoemde task feedback groep. Dit houdt in dat jij feedback krijgt op de minutiae die jij markeert. Wanneer je een groene flash ziet heb jij de minutiae perfect gemarkeerd, niks meer aan doen. Is de flash geel, dan heb je wel het goede punt te pakken, maar heb je het verkeerde label gebruikt, of staat de staart niet in de goede richting. Wanneer de flash rood is heb je helaas geen goed punt te pakken. Wanneer je de feedback ontvangt mag je de markering natuurlijk verwijderen of aanpassen, maar dit is niet verplicht. Wanneer jij denkt dat je toch gelijk hebt kan je hem gewoon laten staan.

Scene 4.2 (Current Feedback):

Als laatste wil ik je nog een snelle uitleg geven over de feedback die jij gaat ontvangen tijdens de eerste twee vergelijkingen. Net als in het echt ontvang je niet veel feedback. Wel krijg je te zien of jouw keuzes bij het kiezen van de patronen, of jouw uiteindelijke mening correct zijn. Mocht je nou niet het juiste antwoord gevonden hebben probeer dan vooral uit te vinden waar het mis is gegaan.

Scene 4.3 (Peer Feedback):

Als laatste wil ik je nog een snelle uitleg geven over de feedback soort die aan jou is toegewezen. Jij behoort namelijk tot de peer feedback groep en dit houdt in dat jij feedback gaat geven aan en krijgen van een ander persoon. Na jou eerste eigen vergelijking ga je iemand anders zijn werk nakijken en zo nodig verbeteren. Met behulp van de notities kan je aangeven waar deze persoon op zou moeten letten bij een volgende vergelijking. Wanneer je dit gedaan hebt ben je klaar met het eerste deel en kun je jouw bestanden opsturen. Het duurt dan maximaal een aantal dagen voordat jij het tweede deel opgestuurd krijgt samen met de feedback die iemand anders aan jou gegeven heeft.

Scene 5:

Mocht alles duidelijk zijn dan kan je nu door naar de volgende pagina, maar mocht je ondertussen wat dingen vergeten zijn dan kan je deze video nog zo vaak kijken als je wilt. Ook kan je sprongen maken in de video door te klikken op de tijdlijn. Als je eenmaal door gaat naar de volgende pagina kan je deze video niet nogmaals bekijken. Veel succes met de training!

Appendix D

The questions asked in the survey after participants finished the experiment.

1. In welke Feedback groep was je geplaatst?
   Dit kan je vinden in de email.

2. Ik vond de feedback handig en had het gevoel dat het me hielp.
   Zat je in de 'NoFeedback' groep, antwoord dan 'oneens'

3. Ik werd zekerder van mijn keuzes naarmate ik verder kwam in de training

4. Ik had het gevoel dat ik meer feedback nodig had dan dat ik kreeg.

5. Ik vond de feedback soms overbodig en/of irritant.
   Zat je in de 'NoFeedback' groep, antwoord dan 'oneens'

6. Wat voor feedback zou je willen toevoegen aan de training?

7. En als laatste… Wat vond je van deze training?

Appendix E

The Dutch consent form and brochure that were handed to the participant before starting the experiment.

Algemene informatie vingerafdruk training

Doormiddel van dit onderzoek proberen we meer inzicht te krijgen in welke soorten feedback nuttig kunnen zijn tijden het bestuderen van vingerafdrukken tijdens de eerste fase van het leerproces. Het doel van het project is namelijk om opstap te creëren voor vervolg projecten die leiden tot een nieuwe manier van vingerafdruk experts opleiden. Dit is belangrijk omdat momenteel de leermethodes in dit domein achterhaald zijn en dit leidt tot problemen voor zowel studenten als begeleiders.

Het doel van dit experiment is om meer inzicht te krijgen in welke feedback soorten veelbelovend kunnen zijn voor het oefenen van vingerafdrukvergelijkingen en daarom vragen wij jou om een online vingerafdruk training te volgen. Om deze training te volgen zal je een Windows programma moeten downloaden die verstuurd zal worden via email. Tijdens deze training ga je leren hoe een vingerafdruk analyse in zijn werk gaat en ga je deze analyse ook zelf uitvoeren. Tijdens deze training worden al jouw keuzes en antwoorden opgeslagen door het programma zodat deze later kunnen worden bestudeerd en er conclusies uit getrokken kunnen worden. Aangezien dit de eerste keer is dat je je gaat verdiepen in vingerafdrukken is het logisch dat je niet alles in een keer goed kan doen. Fouten maken is dus ook helemaal niet erg en voor ons is het juist interessant om te zien waar de knelpunten liggen. Er worden tijdens de training GEEN andere opnames gemaakt zoals scherm-, video- of geluidsopnamen. De training kan je gewoon thuis op je eigen computer maken op een zelf gekozen moment.

De persoonlijke data, die enkel bestaat uit een gebruikersnaam, email en de gegeven antwoorden, zal niet worden gedeeld met derden en alle data zal worden geanonimiseerd voor eventueel gebruik in documenten en/of het publiceren ervan. Er worden verder geen persoonlijke of gevoelige vragen gesteld. Ook mag je je ten alle tijden terug trekken uit de samenwerking zonder daar een verklaring voor af te hoeven leggen. In dat geval zal al jouw data binnen een week worden verwijderd. Echter, wanneer het project is afgerond (2 Juli 2021) kunnen jouw resultaten niet meer uit de gepubliceerde bestanden worden verwijderd. Wanneer jouw data door mij ontvangen is mag je het gedownloade programma verwijderen van jouw computer. Jouw persoonlijke data zal binnen twee maanden na het afronden dit project verwijderd worden.

Mocht je vragen hebben betreft jouw rechten als een deelnemer, of wil je graag een vraag stellen of bedenkingen bespreken met iemand anders dan de onderzoeker, neem dan contact op met de 'Secretary of the Ethics Committee of the Faculty of Electrical Engineering, Mathematics and Computer Science' van Universiteit Twente via ethicscommittee-cis@utwente.nl .

Heeft u vragen over het project en/of het onderzoek, neem dan contact op met de onderzoeker of een van de begeleiders:

Desly Elskamp
d.elskamp@student.utwente.nl
+316 24576038

Job Zwiers
j.zwiers@utwente.nl

Caroline Gibb
c.gibb@utwente.nl

Toestemmingsformulier vingerafdruk training

Deelname aan de studie:

Ik heb de studie informatie gelezen en begrepen op de datum [/ /], of het is mij voorgelezen en ik heb het begrepen. Alle vragen die ik had betreft het onderzoek heb ik kunnen stellen en zijn naar mijn tevredenheid beantwoord.

Ja / Nee

Ik bevestig dat ik vrijwillig deelneem aan dit onderzoek en ik begrijp dat ik me ten alle tijden kan terugtrekken zonder dit te hoeven verklaren.

Ja / Nee

Ik begrijp dat tijdens het onderzoek mijn keuzes en antwoorden automatisch worden opgeslagen en zullen worden gebruikt in het project.

Ja / Nee

Gebruik van data in het project:

Ik begrijp dat mijn data wordt gebruikt om inzicht te krijgen in welke feedback soorten veelbelovend zijn in de educatie van vingerafdruk experts.

Ja / Nee

Ik begrijp dat mijn persoonlijke informatie, dat kan leiden naar mijn identiteit (zoals mijn naam of mijn emailadres), niet zal worden gedeeld met derden en dat mijn data zal worden geanonimiseerd voor gebruik in het project.

Ja / Nee

Ik ga ermee akkoord dat mijn keuzes en antwoorden tijdens de training automatisch worden opgeslagen en worden gebruikt in het project.

Ja / Nee

Handtekening:

___________________________	_________________	___/___/_____
Naam vrijwilliger	Handtekening	Datum

Contact gegevens:

Desly Elskamp
d.elskamp@student.utwente.nl
+316 24576038

Job Zwiers
j.zwiers@utwente.nl

Caroline Gibb
c.gibb@utwente.nl